# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-056841
(43)Date of publication of application : 03.03.1995

(51)Int.Cl. G06F 13/10
G06F 3/14

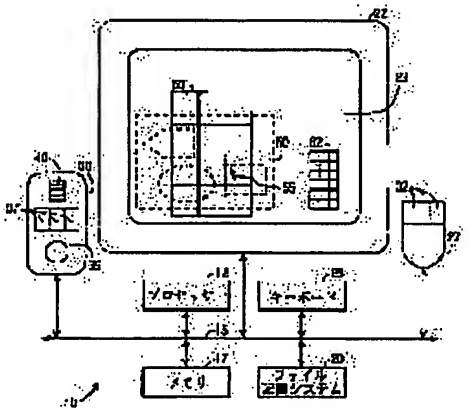
(21)Application number : 06-163458
(22)Date of filing : 15.07.1994

(71)Applicant : XEROX CORP
(72)Inventor : BIER ERIC A
BUXTON WILLIAM A S
STONE MAUREEN C

(30)Priority
Priority number : 93 95445 Priority date : 21.07.1993 Priority country : US

## (54) METHOD FOR OPERATING DEVICE OF PROCESSOR BASE

(57)Abstract:
PURPOSE: To allow a user to operate many general tasks by a fewer operations.
CONSTITUTION: This method is related with a user interface technique to be executed in a device environment controlled by a processor for executing a program for processing a set of element data, and displaying the visible indicator, and the visible display of a set of tools is applied. The tool includes a click through tool, the tool is synthesized with the other tools, and the synthetic tool can be applied. The click through tool generally includes a transparently framed active area, and the area can be moved and placed on the desired part of the visible indicator. When a user interacts through the active area with the visible indicator, the operation applies the characteristic of the specific click through tool. The click through tool can be overlapped on the other tools. In this case, the operations obtained through the both tools on the visible indicator apply the characteristics of the both tools.



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

**特開平7－56841**

(43)公開日　平成７年(1995)３月３日

| (51)Int.Cl.[6] | 識別記号 | 庁内整理番号 | ＦＩ | 技術表示箇所 |
|---|---|---|---|---|
| Ｇ０６Ｆ　13/10 | ３３０　Ｂ | 8133－5Ｂ | | |
| 3/14 | ３４０　Ａ | | | |

審査請求　未請求　請求項の数６　ＯＬ　（全 40 頁）

(21)出願番号　特願平6－163458

(22)出願日　平成６年(1994)７月15日

(31)優先権主張番号　０９５４４５

(32)優先日　1993年７月21日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　590000798
ゼロックス　コーポレイション
ＸＥＲＯＸ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国　ニューヨーク州　14644
ロチェスター　ゼロックス　スクエア
（番地なし）

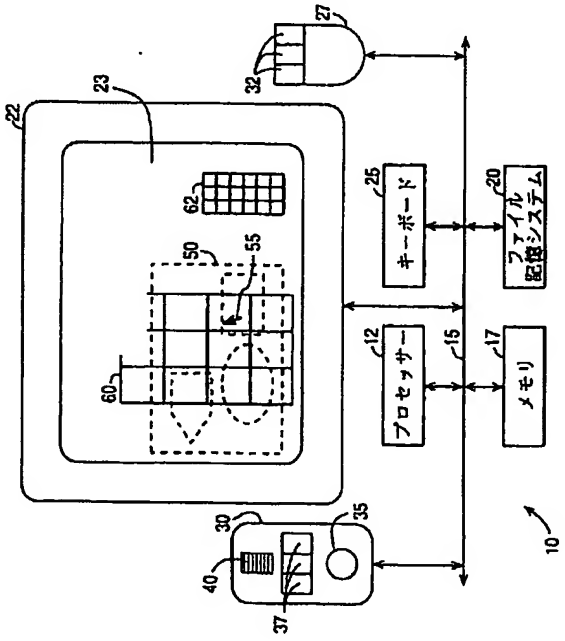
(72)発明者　エリック　エイ．ビヤ
アメリカ合衆国　94043　カリフォルニア
州　マウンテン　ヴュー　シャーランド
アヴェニュー　175

(74)代理人　弁理士　中島　淳　（外２名）

最終頁に続く

(54)【発明の名称】　プロセッサーベースの装置の操作方法

(57)【要約】
【目的】ユーザーがより少ない動作で多くの一般的タスクを行える。
【構成】一組の素データに処理を行いその可視表示体を表示するためのプログラムを実行するためのプロセッサーで制御される装置環境内で実行されるユーザーインターフェイス技法に関し、さらに一組のツールの視覚表示を与える。ツールはクリックスルーツールを含み、該ツールは他のツールと合成されて合成ツールを与えることが可能である。クリックスルーツールは一般的に透明な枠取りされたアクティブ領域を含み、該領域は可動であるため可視表示体の所望の部分上に置かれることが可能である。ユーザーがアクティブ領域を通して可視表示体と対話する場合、動作は特定のクリックスルーツールの特性を与える。クリックスルーツールは他の該ツール上へ重ね合わせられることが可能であり。この場合両ツールを通して可視表示体上にとられた動作は、両ツールの特性を与える。



1

【特許請求の範囲】

【請求項１】　プロセッサーベースの装置の操作方法であって、前記装置はユーザー入力装置と、ディスプレイ装置と、ユーザー入力装置およびディスプレイ装置に結合されるプロセッサーと、命令を含む情報を記憶するための記憶装置を含み、該命令はプロセッサーおよび一連の関連するデータにより実行される少なくとも一つのプログラムを定義し、前記方法は処理ステップを実行するためにプロセッサーを動作させることを含み、該処理ステップはデータの処理を行って可視表示体をディスプレイ装置上に表示するためにプログラムを実行するステップと、複数のツール定義領域の視覚表示をディスプレイ装置に表示するステップと、各ツール定義領域は該領域内のカーソルイベントに応答してデータに行われる特定の動作の少なくとも一部分を指定し、各ツール定義領域およびそれが指定する動作はツールと呼ばれ、クリックスルーツールと呼ばれるツールの少なくとも一つが、データの処理結果がツール定義領域内のカーソルイベントの位置に依存する特性を持ち、第一のツール定義領域を可視表示体に対し置くステップと、少なくとも部分的に第一のツール定義領域とオーバーラップするようクリックスルーツールに相当する第二のツール定義領域を置くステップと、ユーザー入力装置からの一組の信号に応答してカーソルを可視表示体に対し置くステップと、ツール定義領域のオーバーラップ領域内のカーソルイベントに応答して第一および第二のツール定義領域により少なくとも部分的に指定される合成動作と呼ばれる動作を行うステップと、を含むプロセッサーベースの装置の操作方法。

【請求項２】　第一のツールがクリックスルーツールであり、オーバーラップ領域内のカーソルイベントに応答して合成動作がデータに行われ、該合成動作が第一のツールにより指定される動作および第二のツールにより指定される動作を含む請求項１の方法。

【請求項３】　第一のツールが表示物特性を指定するクリックスルーツールであり、第二のツールが特定タイプの表示物の作成を指定するクリックスルーツールであり、表示物が第一のツールにより指定される特性を有している場合に第二のツールにより指定されるタイプの表示物の作成が合成動作の結果となる請求項１の方法。

【請求項４】　第一のツールが第一の表示物特性を指定するクリックスルーツールであり、第二のツールが第二の表示物特性を指定するクリックスルーツールであり、特定の既存表示物を指定する位置情報によりオーバーラップ領域内のカーソルイベントが特徴づけられ、既存の表示物に第一のツールで指定された第一の特性をもたせることおよび第二のツールで指定された第二の特性をもたせることが合成動作の結果となる請求項１の方法。

【請求項５】　第一のツールが動作結果が第一のツール定義領域内のカーソルイベント位置に依存しない特性を有する従来のツール呼ばれるツールであり、第一のツールのツール定義領域内のカーソルイベントの結果が、その後のカーソルイベントの組の結果が特定のタイプの表示物の作成となるモードにカーソルを置き、第二のツールが特定の表示物特性を指定するクリックスルーツールであり、合成動作が、その後のカーソルイベントの組の結果が第二のツールで指定される特性を有する表示物を用いて第一のツールにより指定されるタイプの表示物の作成となるモードにカーソルを置く請求項１の方法。

2

【請求項６】　表示画面と表示画面に関する表示物の位置設定に適した少なくとも一つの入力装置とに結合するプロセッサーと、該プロセッサーが実行し、表示画面の少なくとも一部を制御し、入力装置に応答するユーザーインターフェイスソフトウェアとを備える対話型コンピュータ環境において特定の特性を有するソフトウェアツールをワークピースに適用する方法であって、該方法が、プロセッサー、ユーザーインターフェイスソフトウェア、および表示画面を用いてワークピースを表す内容を有するウィンドウを表示するステップと、プロセッサー、ユーザーインターフェイスソフトウェア、および表示画面を用いて第一のツールを表す第一の透明な表示物および少なくとも部分的に第一のツールとオーバーラップする第二のツールを表す第二の透明な表示物を表示するステップと、プロセッサー、ユーザーインターフェイスソフトウェア、および入力装置を用いてワークピースを表すウィンドウに関してツールのオーバーラップ領域を位置指定することで第一および第二のツールと対話するためにワークピースの少なくとも一部分を選択するステップと、プロセッサー、ユーザーインターフェイスソフトウェア、および入力装置を用いて前記のように選択されたワークピースの一部分へオーバーラップするツールを、ワークピースの選択部分の内容をオーバーラップするツールの少なくとも部分的特性により決定されるように変更するよう選択的に適用するステップと、を含むソフトウェアツールのワークピースへの適用方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明はコンピュータのようなプロセッサーで制御される装置に関し、より詳細には装置とユーザーとの対話を可能とするためのユーザーインターフェイス装置に関する。

【０００２】

【従来の技術】コンピュータのようなプロセッサーで制御される装置の使用は、ユーザへ情報を伝達し、ユーザーから情報を受け取ることをしばしばその目的としている。これはユーザーが特定タスクを実行することを可能にする。これから行おうとするタスクに依り、ユーザーはしばしばワードプロセッサ（時にテキストエディタと呼ばれる）、スプレッドシート、またはドローイングプログラム（時にグラフィクスエディタと呼ばれる）のよ

うな特定タスクアプリケーションプログラムを利用する。多くのプログラムは少なくとも二つ以上の機能を有しているため、特定タイプのプログラムやエディタを引用することは、特定機能のみを有するスタンドアロンアプリケーションプログラムを意味することを意図してはいない。

【０００３】通常のアプリケーションプログラムは、一連の命令（アプリケーション）から成り、該命令は、関連するデータ（時に素データと呼ばれる）の作成や修正のための入力信号に応答して実行される。多くの場合、関連するデータはデータファイル（時にファイルと呼ばれる）としてディスク上に記憶され、各部分がプログラム実行中にメモリへ読み込まれる。少なくともいくつかのアプリケーションについては、データは（例えば印刷またはスクリーン上にディスプレイされる）目視されるドキュメントを表しており、アプリケーションはユーザーがドキュメントを修正することを可能にする。

【０００４】多くの場合ユーザーは、一つまたはそれ以上の入力装置を通して少なくともいくつかの入力信号を与える。該入力装置は、しばしば、キーボードやマウスのようなポインタ装置となっている。マウスは作業表面上を動かされる装置であり、通常キーボードのとなりに置かれ、その動きにしたがったスクリーン上のカーソルの動きを起こす位置信号を与える。カーソルはポインタまたは注意を集める手段として対話プログラムにより使用される特別なシンボルである。マウスは付加的入力信号を与えるための一つもしくはそれ以上のプッシュボタンスイッチ（ボタン）を有しており、該信号はカーソルイベントの一部として翻訳される。

【０００５】ディスプレイ装置は通常、陰極線管(CRT)や液晶ディスプレイ(LCD)のような視覚ディスプレイ装置であり、ユーザーにアプリケーションや素データの情報を与え、ユーザーが適正な入力信号を発生させて所望の処理出力を得るために装置を制御することを可能にする。入力装置、ディスプレイ装置、およびアプリケーションが供給する情報の性質からなる組合せは、アプリケーションに対するユーザーインターフェイスとして考えられてもよい。

【０００６】いづれのアプリケーションプログラムについてもそれが完全に自己充足的なものであることが原理的に可能であるが、アプリケーションプログラムはほとんどすべての場合においてオペレーティングシステム(OS)と共に処理を実行する。OSは装置リソースを管理および制御してアプリケーションプログラムと装置ハードウエア間のインターフェイスを可能にするプログラムである。OSは通常、基本的ハウスキーピング機能を提供する。該機能はすべてのアプリケーションプログラムが必要とするものであり、ファイルシステムのメインテナンス、CPU の管理、入力装置からの入力の受信、記憶装置との通信、ディスプレイ装置へのデータの送信、およびユーザーがファイル管理や種々のアプリケーションプログラムを実行出来るように一般的メカニズムを供給すること等である。パーソナルコンピュータ(PC)およびワークステーションの世界において、オペレーティングシステムはしばしば特定タイプのハードウエア構成と関係づけられるが、必ずしもそうである必要はない。多くのタイプの装置上で過去から現在にかけてなお走り続けているUnixがこの例である。

【０００７】近年その使用が増加しつつあるオペレーティングシステムの一つのタイプに、グラフィカルユーザーインターフェイス("GUI")を提供するものがある。AppleComputer社のMacintosh OS、IBM 社のOS/2、およびMicrosoft 社のWindows （実際には、DOS として知られているキャラクタベースのオペレーティングシステムの最上位階層で走るGUI シェル）は、PC界で知られる最良のGUI である。Macintosh OSは、Motorola 680x0ファミリーのマイクロプロセッサを基本とするApple社製のPC上でのみ使用可能であり、一方、OS/2およびWindowsは、Intel 80x86ファミリーのマイクロプロセッサを基本とするいわゆるIBM PC上でのみ使用可能である。二つのタイプ以上のマイクロプロセッサ上で走ることが可能なバージョンを有するMicrosoft 社のWindows NTにより、この傾向が変わりつつある。

【０００８】GUI に関する一つの性質は、アプリケーションに対するオープンファイルにスクリーン上の可動およびサイズ変更可能領域であるウィンドウが通常与えられることである。OSはディレクトリ構造を示す自身のウィンドウを有することが可能であり、OSが伴うファイルやアプリケーションはアイコン（働きや項目を表す小さい図形表示物）により表されることが可能である。オープンファイルに対応しない他のウィンドウがあってもよい。GUI の利点はアプリケーションに交わるかなり一貫したユーザー環境を提供することである。いくつかのGUI は複数のアプリケーションを同時に開くことを可能とする。

【０００９】OSのタイプによらずアプリケーションプログラムは通常、OSの多くの支援を伴い、素データの表示体（時にスクリーンイメージまたはディスプレイイメージと呼ばれる）をユーザーに提供する。ユーザーは該表示体に働きかけ、プログラムがこれらのはたらきかけを素データの処理に翻訳する。本文中において表示体という言葉は、素データだけでなくOSや様々なタイプのユーティリティープログラムを含んだすべての種類のプログラムの表示体に関する。

【００１０】例えばワードプロセッサにおいて素データはテキストから成り、該テキストはドキュメントがプリンタ上にプリントアウトされる時にそれがどのように見えるかを特定する関連情報を伴っている。該関連情報はパラグラフやコラム、およびフォント、サイズ、スタイル、色等のテキスト形態のようなドキュメントのレイア

ウトに関している。特定のワードプロセッサおよびオペレーティングシステムに依り、スクリーンイメージはテキスト内容に限定されてもよいし、またはそれがプリントされる時に現れるドキュメントを実質的に示してもよい。（WYSIWYG;What you see is what you get; 見たものがそのまま得られる）DOS のようなキャラクターベースのOSのために設計されたプログラムは前者に近いものを提供する傾向にあり、GUI のためのそれは後者に近いものを提供する傾向にある。

【００１１】他のタイプのアプリケーションプログラムにおいても同様な範囲のスクリーンイメージを見出すことが可能である。例えばドローイングプログラムにおいて、素データはドキュメント上に現れる各図形表示物（オブジェクト）の記述を有している。該記述は図形表示物に所望の外見を与えるために必要な項目を含んでおり、該項目は、形、サイズ、ライン色および幅、充填色およびパターン、ドキュメント平面上の相対位置、およびスタック順序（目的の図形表示物が他の図形表示物の上面か背面かどうか）等を含む。スクリーンイメージは、図形表示物の輪郭のみ（ワイヤーフレーム図）を示してもよく、または、完全なWYSIWYG 図であってもよい。

【００１２】アプリケーションのタイプによらずユーザーは所望の変化を得るようスクリーンイメージを参照しながら入力装置を操作する。これは通常、修正対象表示物の表示位置に対応するスクリーン上の特定の位置にカーソルを置くこと、およびキーストロークやマウス動作のような一つまたはそれ以上のユーザーイベントの実行によりなされる。マウス動作はボタンを押すこと、ボタンをはなすこと、マウスの移動、クリック、およびドラッグを含む。マウスクリックはユーザーがマウスを止めたままで一つのボタンを押してはなすことであるが、この言葉はまた一つのボタンを押す動作を言い表すことにも使われる。ドラッグ（時にクリックドラッグ）はユーザーがマウスでカーソルの位置設定を行うことであり、一つのボタンを押し、ボタンを押したままマウスを新しい位置に動かし、新しい位置でボタンをはなすことである。マウスボタンを押すこと、ボタンをはなすこと、クリック、およびドラッグはキーボードのキーまたは他のマウスボタンを押したまま（もし現状がそうであれば）行われてもよい。

【００１３】例えばワードプロセッサーイメージ（画像）においてカーソルを特定の位置に置くことは、タイプ入力されたテキストをその位置に挿入するために行われてもよい。表示されたテキストの一部にわたるカーソルドラッグは（この時スクリーン上に強調表示される）テキストの選択であってもよく、これにより選択されたテキストに対しユーザーが他のいくつかのメカニズムによる（削除、移動、またはフォント変更のような）処理を行うことが可能となる。アプリケーションおよび所望の処理に依り、該メカニズムはメニューからの処理の選択、またはキーボードからのコマンド入力であってもよい。

【００１４】ドローイングプログラムにおいても同様にクリック動作によりカーソルがツールアイコン（例えば矩形ツール、ラインツール、多角形ツール等）上に置かれることが可能であり、この後のカーソルクリックおよびドラッグにより図形表示物が作成されることが可能となる。すでに存在している図形表示物上での平面カーソルクリックは、他のいくつかのメカニズムを通して処理が行われるよう該表示物を選択することであってもよい。カーソルドラッグが図形表示物上から始まる場合、ドラッグの結果はカーソルの動きに沿った該表示物の移動を起こしてもよいし、または該表示物上のカーソル位置に依る該表示物のサイズ変更を起こしてもよい。

【００１５】

【発明が解決しようとする課題】ユーザーをより生産的にするためユーザーには比較的容易に学習でき、容易に使用でき、強力であるツールが提供されるべきである。これらの課題は時に個別には容易に達成できるが、組み合わせた場合はまれである。それにもかかわらずより直覚的、効率的、融通のきくユーザーインターフェイスを設計する試みに対しかなりの労力が費やされてきた。以下ドローイングプログラムを例にとり、これまでどのように労力が使われてきたか、およびユーザーインターフェイスの一特徴を改善することが他の特徴を低下させてしまうことがありえることについての説明を行う。

【００１６】ドローイングプログラムの一般的構成は、ドローイング（描画）領域の片側に固定されたツールパレットおよびドローイング領域上部のメニューバーを有する。ツールを変更するためユーザーはカーソルをパレットへ動かし、所望ツールアイコンをクリックし、そして再びカーソルをドローイング領域内の適正位置へ動かす。所望の表示物上に所望の処理を施すため、ユーザーはカーソルを表示物へ動かし、表示物をクリックして該表示物を選択し、カーソルをメニューバーへ動かし、マウスボタンを押してメニューを引き出し、所望のメニュー項目をドラッグし、そしてマウスボタンをはなす。この後ユーザーはドローイング領域か、メニューバーの他の項目か、またはツールパレットへカーソルを動かす。これはたとえ最も単純な処理に対したとしても多くのマウス移動となる。

【００１７】ティーオフ（切取り）メニューおよび可動ツールパレットは、作図が実際に行われている領域付近にユーザーが一定量のメニューとツールパレットを継続して開いておくことを可能とし、その結果マウスの移動距離を減少させる。ティーオフメニューおよび可動ツールパレットはユーザーのカーソル移動距離を減少させる点ではより効率的作図をもたらすが、他の一方ではより非効率的なことをもたらす。これらは特にユーザーが作

図している付近の多くの作図領域を占める傾向にある。このことはメニューとパレットを作図のじゃまにならないように移動するためにユーザーが作図作業を常に中断しなければならない結果となりえる。プログラムが強力（機能的）になればなるほどメニューがながく伸びてより多くの領域を占めるという事実によりこの問題は複雑化する。前述の課題を解決しようとする際に遭遇するこの種のトレードオフは不幸にしてまれなものではない。

【００１８】

【課題を解決するための手段】本発明はユーザーがより少ない動作で多くの一般的タスクを行うことを可能とし、その結果として生産性を高めるユーザーインターフェイスの技法を提供する。該技法は、ユーザーが親しみやすく、それゆえかなり速く学習することも可能であるような動作を利用する。本発明は単一プログラム中で実施されてもよいし、またはオペレーティングシステムを含む異なるプログラム間にわたって有効となるようオペレーティングシステムに組み込まれてもよい。

【００１９】本発明はプロセッサーで制御される装置環境内で動作し、該装置は一組の素データの処理をして可視表示体を表示するプログラムを実行する。さらに本発明に関するシステムは、ツールパレット、特性パレット、メニュー、スイッチ、ダイアログボックス、およびスライダーのような一組のコントローラーの視覚表示を与える。コントローラーは集合的にツールと呼ばれ、ユーザーは通常（一組の入力装置を使用して）ツールおよび可視表示体との対話によりプログラムに処理を指定し、ディスプレイ装置上でこれらの処理結果を見る。ユーザーは通常、カーソルを所望の位置に置き、カーソルを用いてマウスクリックのような動作を行う。

【００２０】簡潔には、本発明はクリックスルーツールと呼ばれるもので特徴づけられ、該ツールは（他のクリックスルーツールを含む）他のツールと合成されて合成ツールを与えることが可能である。クリックスルーツールの視覚表示は一般的に透明な枠取りされたアクティブ領域を含み、該領域は可動であるため可視表示体の所望の部分上に置かれることが可能である。ユーザーが該アクティブ領域を通して可視表示体と対話を行う場合、特定のクリックスルーツールの特性についての動作が行われる。可視表示体が一組の図形表示物を表すものとなるドローイングプログラム中で動作する本発明の実施例では、クリックスルーツールは表示物の作成のためのツールおよび既存の表示物のコピー、変更、および削除のためのツールを含んでもよい。

【００２１】クリックスルーツールが可視表示体に対していったん置かれると、該ツールを通しての動作結果は通常、動作が起きたアクティブ領域内の特定の位置に依存する。他の方法を用いると、クリックスルーツールを通しての動作結果が通常、動作が起きた時点のカーソル下の可視表示体の一部分に依存する。従来ツールもまたアクティブ領域を有しているが、ツール上の動作結果は該ツールに依存し、該結果は通常、動作が起きたアクティブ領域内の位置または可視表示体に対するツールの位置に依存しない。従来ツールは、そうあらなければならない基本的理由がないにもかかわらず、通常は透明でない。

【００２２】クリックスルーツールと他のツールを結びつける動作は合成と呼ばれる。クリックスルーツールは他のツールと重ね合わされることが可能であり、この場合二つのクリックスルーツールを通して可視表示体上に行われた動作は両ツールの特性を与える。二つのツールは、もし必要なら単一のものとして動くようにグループ化されることが可能である。クリックスルーツールは従来のツール上に重ね合わされることも可能であり、クリックスルーツールを通して従来のツール上へ動作が与えられる。同様に、この動作は両ツールの特性を与える。従来ツールが状態ツールである場合、状態ツールを用いた結果として生じる動作は、クリックスルーツールの特性をもつものとなる。加えて、従来の状態ツールが呼び出されてその後の動作がクリックスルーツールを通して行われることが可能であり、この場合従来ツールの使用はクリックスルーツールの特性を与えるものとなる。クリックスルーツールが他のツールの一部分上のみに重ね合わされる場合、ユーザーはオーバーラップしている領域内をクリックして合成動作を行うこと、またはオーバーラップしていない領域内をクリックして単位な合成されていない動作を行うことの選択が可能となる。

【００２３】特定の実施では各クリックスルーツールは命令を与え、ツールがイベントを受け取った時点までに該イベントが累積した動作とツールの動作とを合成する役目を負う。様々な方法の内の一つを用いて既存の動作から新たな動作が算出されることが可能であり、該方法は動作の命令リストを既存の命令リストの終端に追加して合成動作を作成すること、動作の命令リストを既存の命令リストの先頭に追加すること、命令リストからひとつまたはそれ以上の命令を削除すること、受信した動作のいくつかまたはすべてで使用されている命令表記を変更すること、命令の命令表記に続き指定された論理値を変更すること、または命令内に指定された座標<x, y> を変更することを含む。いくつかの実施例ではツールの集合がユーザーの制御下で共に動き、それゆえ該集合は可視表示に対して動かすことの可能なオーバーレイ上に置かれていると考えることが可能である。オーバーレイは、ユーザーの利き腕でない方の手（例えば右利きのユーザーの左手）により制御される入力装置により位置設定されることが好ましい。

【００２４】

【実施例】

１．０　システム外観

図１は、本発明の実施例であるコンピュータシステム１

０を示す。従来技術に従い該システムは、バスシステム１５を通して多くの周辺装置と通信を行うプロセッサー１２を含む。これら周辺装置は通常、メモリ１７およびファイル記憶システム２０を含んだ記憶装置、多くの入力装置、およびアクティブ表示領域２３を有するディスプレイ装置２２を含む。ファイル記憶システムはプログラムとデータファイルを記憶し、ハードディスクドライブやフロッピーディスクドライブのような標準装置を通常含む。該装置はCD-ROMドライブや光学的ドライブ装置のような他の装置を含んでもよい。

【００２５】本文中においてバスシステムという言葉は、システムの各構成装置に所望の双方向通信を行わせるためのあらゆるメカニズムを含ませるために一般的に用いられる。入力装置とディスプレイ装置を除き、他の構成装置は物理的に同位置にある必要はない。例えばファイル記憶システムの一部は様々な長距離ネットワークメディアを通して接続されることが可能である。本発明はほとんどPCやワークステーションのプログラムにおいて実施されることが予想されるが、前記と同様に入力装置とディスプレイ装置はプロセッサーと同位置にある必要はない。

【００２６】入力装置はほとんどの部分が標準装置であり、キーボード２５と一つまたはそれ以上のポインタ装置を含む。マウス２７とトラックボール３０が示されているが、タッチスクリーン、グラフィックタブレットまたは電子スタイラスが用いられてもよい。従来システムにおいて二つ以上のポインタ装置が用いられる場合があるかもしれないが、ユーザーは作業時に一つの該装置のみを使用するのが通例である。本発明はユーザーに二つの該装置を提供することで重要な利点を得るものであり、該装置は両手に提供され、同時にまたは交互に使用される。具体性を持たせるためマウス２７は三つのボタン３２を有するよう示されており、トラックボール３０はボール３５、三つのボタン３７、およびサムホイール４０を有するよう示されている。

【００２７】以下本発明について、図に示された表示領域２３の内容を参照し、高度なユーザーの観点から説明を行う。ディスプレイ装置はドローイングプログラム用の一つめのアプリケーションウィンドウ５０とワードプロセッサー用の二つめのアプリケーションウィンドウ５２を表示している。ドローイングウィンドウは三つの図形表示物である、矩形、楕円、五角形を有するよう示されている。ワードプロセッサーウィンドウはテキストを有するよう示されている。その位置をマウス２７により制御される矢印の形のカーソル５５がドローイングウィンドウ内の矩形の外郭線上に位置するよう示されており、これは所望の処理を行うためにユーザーが該矩形を選択した場合であってもよい。これは代表的設定であり、例えばユーザーが特許明細書とそれに関する図面を作成しているところで起こる例であってもよい。コンピュータとこれから行われるタスクに依り、表示領域全体を一つのウィンドウが占めていてもよく、また、多数のウィンドウがオーバーラップして占めていてもよい。

【００２８】前述のコンピュータ環境および表示領域の内容は標準的なものとなっている。本発明は該環境に他の一態様を付加する。該付加態様は多数の枠取りされた領域６０を有する可動透明オーバーレイである。該枠取り領域はマルチエレメントグリッド内の互いに境界を接する複数の矩形として示されているが、後述のように必ずしも互いに境界を接する必要はない。さらに、必ずしも同時に複数がディスプレイ上に表示される必要はない。オーバーレイ上の枠取り領域を表示領域内にある表示物と区別するための補助として、オーバーレイ上の表示物が実線で示され、アプリケーションウィンドウと図形表示物が点線で示されている。後述のように、オーバーレイは特定される枠取り領域の重要性を指定するための（アイコンまたはテキストのような）指標を伴うことが望ましい。このためオーバーレイは透明なものとして与えられる一方で、オーバーレイ上の枠取り領域の必要からいくつかの不透明部または半透明部を有してもよいと考えられる。

【００２９】所定の枠取り領域が表示の一部上に置かれ、該領域内で動作が行われる場合、該動作は特定された該領域の特性を持つものとなる。このように各枠取り領域は、表示領域の関連部分に運ばれて該部分に適用されることが可能となるツールのアクティブ領域と考えられてもよい。このようなツールの適用され方の性質が与えられることから、該ツールは時にクリックスルーツールと呼ばれる。後述の多くの説明はオーバーレイを単一な透明シートとして扱っているが、オーバーレイは、多くの半透明ツールをそれぞれ有し相対的に可動な複数の透明シートの形態を含んでもよい。

【００３０】適用対象表示物上にツールを現すことは最も直覚的アプローチであると考えられ、また該アプローチは一般的に認められるものである。しかしこれとは逆のスタック順序を保証する特別な環境があってもよい。例えば、オーバーレイ上のマークによってでさえいかなるアプリケーション表示物も不鮮明であってはならないようなクリティカルなアプリケーションがあってもよい。この場合にはアプリケーションを透明に現し、オーバーレイをアプリケーションの背面に現すことによりアプリケーションが可能となる。後述のように、オーバーレイプログラムの基本的操作は前記いずれについても同様となる。このためオーバーレイという言葉は、それが表示領域の表示物の上面または背面に現れることにかかわりなく、ツール関係シートの集合を表すために用いられる。いくつかの場合においてはあるスタック順序から他のスタック順序へユーザーが切替えを行うことが可能であるようにすることが好ましい場合もある。

【００３１】ユーザーが表示領域に対してオーバーレイ

を置く方法はたくさんあるが、これはトラックボール３０を用いてユーザーの利き腕でないほうの手により行われることが好ましい。直線運動的位置設定はボール３５の回転により行われてもよく、この他の操作はボタン３７を用いて行われてもよい。オーバーレイおよびその内容の変更はサムホイール４０の回転により行われてもよい。

【００３２】クリックスルーツールとオーバーレイは新たなユーザーインターフェイス素子を提示するが、該素子は標準インターフェイス素子と共に用いられてもよい。従来技術プログラムの多くに使用されるタイプの定型的ツールパレット６２が例として示されている。プログラムとOSに依り、与えられたプログラムに特有のツールとパレットはプログラムウィンドウにしっかり固定されてもよいし、または該プログラムのための他のウィンドウに対し相対的に可動な別個のウィンドウとして現れてもよい。オーバーレイツールの実施例に関する後述の説明の大部分はクリックスルーツールを扱うが、パレット６２内にあるような従来ツールがオーバーレイ上に結合され、該オーバーレイ上の他のツールと共に動かされてもよい。パレット６２は実線で示されており、これは該パレットがオーバーレイ上にあることを示している。従来ツールは一つまたはそれ以上のオーバーレイシートをクリックスルーツールと共有することが可能であり、また別個のオーバーレイシート上に隔離されることも可能である。

【００３３】図２は、メモリ１７またはファイル記憶システム２０に記憶された各データ項目が表示領域２３に現れるよう処理される過程を表すフローダイヤグラムを示す。プログラムの素データ７０は通常、各プログラム特有のフォーマットで記憶されている。該フォーマットはプログラムの特性であり、プログラム操作のためにほぼ最適化されている。データは翻訳装置（または翻訳プログラム）７２による処理に従い、該装置（またはプログラム）はデータをディスプレイ上に現れるべき表示を特定する画像データ構造７３に変換する。フォーマットは多数あり、例えば画像データ構造７３はビットマップであってもよいし、また、Display PostscriptやQuickdraw のようなプログラム言語の一連の命令であってもよい。詳細はともかくとして、画像データ構造は（まだビットマップになっていない場合）ディスプレイの解像度において表示が可能となるよう充分な情報を有していなければならない。そうでない場合にはディスプレイ上の表示のための処理が加えられる。

【００３４】オーバーレイも同様な階層構造により特性づけられる。該階層構造においてオーバーレイの素データ７５は、該データをオーバーレイ画像データ構造８０に変換する翻訳装置（または翻訳プログラム）７７により処理される。前記二つの画像データ構造は結合ノード８２で示される地点で結合され、最終的ディスプレイイメージ８３に変換される。画像データ構造を結合するために特定される技法は、オーバーレイが不透明部または部分的不透明な指標を伴う透明シートとして現れることを保証しなければならない。後述のようにオーバーレイはビジュアルフィルタと呼ばれるようなものを含んでもよいし、またビジュアルフィルタを組み込むツールを含んでもよい。そのような具体例において、結合ノードはディスプレイイメージの各部を修正したりフィルタ処理することが可能である。オーバーレイの構成および外観の仕様について、各ツールの具体例と共に以下詳細な説明を行う。

【００３５】本発明のシースルーインターフェイスは三つの概念的ユーザーインターフェイス層であるカーソル層、オーバーレイ層、アプリケーション層の相対的位置設定を必要とする。カーソル層はその最小限として、平面内の識別された点（カーソル位置）により定義され、該識別点と共に正確に動く一つまたはそれ以上の表示物を含む。オーバーレイ層はその最小限として、共に組み合わされて動く一組のツールを含む。アプリケーション層は表示体を伴う一つまたはそれ以上のプログラムを含む。これら各層は順にサブレイヤーを含んでもよい。例えばカーソルはドラッグドロップ表示物を伴ってもよいし、オーバーレイのツールは単純なツールを互いに積み重ねることで作られてもよいし、また各アプリケーションはオーバーラップウィンドウシステムのようにオーバーラップしてもよい。

【００３６】図３は、これら三層間の関係およびオーバーレイ８５とアプリケーションプログラム８７、８８（アプリケーション＃１、＃２として示されている）との間の情報のやりとりのフロー図を示す。オーバーレイツールが表示体の上面に現れるか背面に現れるかに関係なく、オーバーレイとアプリケーション間の情報のやりとりは同一となる。

【００３７】トリガーがかかるとオーバーレイツールはアプリケーションへ命令を送出する。該命令は任意のデータ構造を含んでもよい。アプリケーションは自身のデータ構造を変えることで該命令に応答してもよいし、オーバーレイへデータを送り帰すことで応答してもよい。加えて、自身をペイントする時はいつでも、アプリケーションは自身の表示体（スクリーン上の現行表示）に関する情報を与えることでオーバーレイに応答する。オーバーレイは該表示体を（例えばビジュアルフィルターを用いることなどによって）ユーザに提示する前に修正してもよい。

【００３８】アプリケーションは一定のコマンドに応答してオーバーレイへデータを送り帰すため、図はアプリケーションからオーバーレイへの前記とは逆の経路も示している。本発明の詳細な実施例に関する後述の説明はクリックスルーツールを扱うが、その基本的内容はオーバーレイ上の従来ツールにも適用される。

【００３９】オーバーレイソフトウェアはウィンドウマネージャー９２と共に処理を実行する。該ウィンドウマネージャーはオペレーティングシステムの一部であってもよく、ウィンドウフレームを作成してもよいし、ディスプレイ上のウィンドウの作成、移動、サイズ変更および記述を監督してもよい。ウィンドウマネージャーは入力信号を入力装置からそのまま受け取り、該信号を適正なアプリケーション（通常はカーソル直下のウィンドウを有するそれ）へ送り、位置情報をアプリケーションの座標系で表された座標に翻訳する。ウィンドウマネージャーはまた、ウィンドウの内容をどこに描くかの情報をアプリケーションに与える。

【００４０】オーバーレイへの入力信号は（例えばマウスイベント座標のような）OSからの入力信号そのままであってもよいし、ドラッグドロップ対象表示物または他のアプリケーションにより与えられてもよい。加えて、オーバーレイツールの重ね合わせが可能であるような実施例については、他のオーバーレイとして見えるものから入力信号が来てもよい。オーバーレイへの一組の付加入力信号（図には明示されていない）は、表示体に対するオーバーレイの位置設定のための信号を含んでもよい。

【００４１】図の実施例においては入力信号は汎用言語の命令に翻訳され、該命令は適正なアプリケーションのための翻訳装置（またはプログラム）へ送出される。入力信号がアプリケーション＃１へ送出される命令を起こす位置情報を有していた場合、該命令は翻訳装置（またはプログラム）９３へ送出される。該翻訳装置（またはプログラム）は該命令のうちのいくつかをアプリケーション＃１の入力言語の命令へ変換し、また該命令のうちのいくつかをアプリケーション＃１の処理呼出しへ直接変換する。アプリケーション入力言語の命令は、該命令を処理呼出しへ変換する命令解析装置（またはプログラム）９５へ送出される。図は入力信号がアプリケーション＃２に関係する場合についても示しており、この場合汎用言語の命令は翻訳装置（またはプログラム）９７へ送出され、次にアプリケーション＃２の処理呼出しを発生させる命令は命令解析装置（またはプログラム）９８へ送出されることが可能となる。

【００４２】汎用言語の命令を前記両アプリケーション言語の命令に変換する一実施例について以下に説明を行う。ビットマップの処理をするペイントプログラムおよび座標指定された表示物の処理をするドロープログラムの実施例を考える。さらに、カーソル下のオーバーレイツールが色を赤に変更することを指定する場合を考える。通常、命令は一連の位置情報である演算子を含んでおり、また、一つまたはそれ以上のパラメータを含むことが可能である。汎用言語の命令が含む実施例として以下を想定することが可能である。

【００４３】SetColor<x, y>red,
SelectCorner<x, y>, and
Scale<x, y><x', y'>2.0.

前記SetColor命令について考える。ピクセルの処理をするペイントプログラムにとって、カーソル位置は必要とされる動作を決定するすべての位置情報を与え、また単一命令が必要とされるすべてである。関連するペイントプログラム言語の単一命令として以下が想定されてもよい。

【００４４】SetColorPixel<x, y>red.

ドロープログラムにとってどの表示物が選択されたかをカーソル位置に基づき決定することが第一に必要であることと想定され、この後該表示物に対し色が指定される。関連するドロープログラム言語の命令シーケンスとして以下が想定されてもよい。

【００４５】SelectObject<x, y>
SetColorSelectedShape red.

選択された表示物の色設定処理がオーバーレイ上の従来ツールで行われた場合、命令シーケンスは同一になることが想定されるが、該命令シーケンスは二つのステップに分かれることが想定される。第一ステップはユーザーが従来方法で表示物を選択することであり、その次のステップはユーザーが従来のカラーパレット内の赤いボタンをクリックすることである。

【００４６】前記設定における（クリックスルーツールを用いた場合に対する）変化事項は、入力信号が汎用言語の命令へ変換されることを避けるためにオーバーレイとアプリケーション翻訳装置（またはプログラム）をしっかり結合させておくこととなることが想定される。さらに、オーバーレイはそれがどのアプリケーションをサポートしているかについての情報を維持しなければならないこと、および入力信号を適正なアプリケーション入力言語に変換することが想定される。

２．０　オーバーレイツール実施例概観

オーバーレイが有用なものとなるために、オーバーレイはユーザーのアプリケーションの利用を支援する一組のツールを有していなければならない。多くの該ツールについて以下に説明を行う。これらツールのいくつかはそれのみで本発明に関するものであり、またそれら以外はオーバーレイの内容物としてのみ本発明に関する。利き腕でない方の手でできるタスクのほとんどは、現行タスクを中断する代償を払えば利き腕によってもなされることが可能である。しかし両手使用を必要とするいくつかのタスクがある。後述の説明のツールの多くはクリックスルーツールとなっている。この言葉は、前述のように表示体の一部上をツールを通してクリックすることでツールが適用されることを表している。

【００４７】該ツールは以下の内容を含む多くの興味深い特性を有している。該ツールはいくつかの対話ステップを一つにまとめることをしばしば可能とする。ユーザーの目は作業領域から離れる必要がなくなる。インター

フェイスは直接的、視覚的、また注意深く選択されたツールを用いることで簡単に学習できるものとなる。ユーザーの利き腕でない方の手は大まかな位置設定にのみ寄与し、微調整はマウスをもつ手で行われる。実施例は画面内の図形表示物を作成、修正、および削除するためのツールを伴うドローイングプログラム（グラフィカルエディタ）環境を基本的に意図している。

【００４８】ほとんどのツールの説明は図面を用いて行われ、用いられる図面は異なる処理段階での作図場面の外観、およびいくつかの場合にはツールのそれを示す一連の描写を含む。いくつかの実施例については、本発明のツールを用いた特定の処理が従来のドローイングプログラムツールと技法を用いた同一の処理と対比される。図８、１３および２３を除き、画面内の表示物は点線で描かれ、またオーバーレイツールは実線で描かれる。これは図１の取決めと逆なものとなる。

【００４９】特定タイプのプログラムやエディタの引用は、スタンドアロンアプリケーションプログラムを意味することを意図してはいない。実際、いわゆるドローイングプログラムの多くは非常に高度なテキスト処理能力を有しているし、またいわゆるワードプロセッサーの多くは強力なドローイングモジュールを有している。単一プログラムで多くのタイプのプログラムの機能を提供する集積プログラムパッケージ（いわゆるワークスプログラム）により、プログラムの区分けはさらにあいまいなもとのとなる。従って特定タイプのプログラムの引用は、明確に規定された機能を有するプログラムの引用としてとらえられるべきであり、該プログラムがドローイングプログラム、ワードプロセッサー、データベースプログラム、またはスプレッドシート等として市販されていることによらない。

【００５０】多くのツールはスナップドラッギングと呼ばれる特徴をもつグラフィカルエディタと共に説明が行われる。これはBierとStone の文献（文献Bier86）にスナップドラッギングとして記載されている（ある表示物が他の表示物を引きつける）引力技法を引用したものである。該技法ではキャレット（脱字記号）と呼ばれる特別な点が表示物の角のような引力が与えられた位置に取り付き、他の表示物が作成されると該表示物がキャレットに取りつくことが可能となる。

【００５１】ボタン、メニュー、パレットという言葉が多くのツールと共に後述の説明で用いられる。これらの言葉は一般に知られている意味で用いられるが、本発明のオーバーレイがこれらに新たな特性を与えて既知のものとは別の手段にしてしまうため、一般的意味からの多少の逸脱が時に必要となる。一般にボタンはディスプレイ上の一つの定義された領域を言い、これがクリックされる時に所望の処理が起こる。オーバーレイで用いられるいくつかのボタンは、ボタン上の特定位置をクリックすることでユーザーが特定の結果を得ることを可能にする。一般にメニュー（しばしばプルダウンまたはポップアップが前に付けられる）は項目または特性のリストであり、メニューバーまたはメニューアイコンをクリックして所望の項目にドラッグすることでユーザーが選択できるものである。パレットという言葉はボタンの集合表示を言い、そこでは一つまたはそれ以上のボタンの選択がクリックにより可能である。

【００５２】ティーオフメニューは実質的にプルダウンまたはポップアップメニューのための代用パレットである。従ってメニュー選択はメニュー項目選択の単一ステップからなり、メニューバーのメニューを選択してからメニュー項目を選択する複合ステップを取る必要はない。後述の説明で用いられるパレットメニューという言葉はパレットまたはティーオフメニューを表し、利き腕でない方の手により動かすことができ、ユーザーを現行の主要タスクからそらすことなく作業領域に持ち込まれ、その後取り去られるものである。

【００５３】後述の特定ツールのいくつかは、ビジュアルフィルタ、フィルタまたはレンズと呼ばれるものを利用する。各フィルタは目視領域と呼ばれるスクリーン領域であり、該領域内に表示された図形に対しワイヤーフレーム形式の拡大、透視、およびスップレッドシートセル内の隠れた同一物の表示のような処理を行う作用素子を伴う。これらフィルタは、ピクセル以外の多くの表示形態、および拡大以外の多くの処理に対し汎用性を与えるものである。表示出力を生成するために、これらフィルタは現行の表示体を生成しているオリジナルなアプリケーションデータ構造を利用する。従ってこれらフィルタはアプリケーションデータ構造を実質的に異なるフォーマットで表すことが可能であり、これにより従来そうすることが困難であった情報の強調表示、現行処理に関係しない情報の圧縮、または従来表示されることのなかったデータ構造の部分的情報でさえ表示することが可能となる。このようなビジュアルフィルタはオーバーレイツール、特に該ビジュアルフィルタにより表示または強調表示されるデータ構造の一部に関わる処理を行うツールと協調して動作する。

【００５４】ビジュアルフィルタはアプリケーションデータ構造の修正表示だけでなく、特定されたアプリケーション図形に対し置かれる一時的オーバーレイツールを生成してもよい。ユーザーはこれら一時的オーバーレイツールをオーバーレイ上の他のツールと同様な方法で使用することが可能である。例えばこれらツールはボタンを含んでもよく、ユーザーはボタンの背面にあるアプリケーションに送出されるべき命令を起こすために該ボタンをクリック、クリックスルー、またはドラッグすることが可能となる。

【００５５】いくつかのフィルタが組み合わされて表示物に順に処理を行う場合、表示物がフィルタの層を下から上へ逐次的に通過するように見える効果を生じる。加

えて、あるフィルタがその下に他のフィルタを有する場合、これら他のフィルタがスクリーン上に与えた境界を該フィルタ自身の境界内に修正してもよい。

２．０１　画面内への表示物の挿入および図形の作成

図４は、オーバーレイ上の図形パレットを用いて新たな図形をグラフィカル画面に加える方法を示す。ユーザーはツール上の円をグラフィカル画面内の矩形付近に大まかに位置設定したところである。ユーザーがマウスボタンを押してその状態を保つと、ツールと同サイズの新たな円が画面内に生成され、オーバーレイが消え、位置設定の微調のために円はその中心を（例えば）カーソルの矢に取り付ける。スナップドラッギング（文献Bier86）のような引力技法を用いることでその中心が正確に矩形の角上になるよう新たな円が置かれることが可能である。ユーザーがマウスボタンをはなすと新しい図形はその最終的位置となり、ツールが再び現れる。ユーザーが三角形のようないくつかの角を有する図形を置いた場合、マウスボタンが押された時点でカーソルにいちばん近い角が該図形をカーソルに取り付ける点となったことが想定される。

【００５６】前記の実施例では、メニュー内のサイズが、それがアプリケーションに適用された時点で、作成図形のサイズを決定している。線、矩形、円、および他の図形の選択の場合のような多くの場面では、ユーザーは図形の一般的形を選択してからサイズや位置を特定することを望む。オーバーレイは、両手を用いて選択、位置設定、およびサイズ設定タスクを流暢で自然な形態で行うことができる利点を有する新たな技法を可能とする。

【００５７】図５は、矩形の四つの角すべてを同時に置くことを可能とする矩形作成の実施例を示す。最初に、利き腕でない方の手が画面内の矩形上にツールの矩形を置いたところである。ユーザーはマウスカーソルでツール矩形をクリックし、マウスボタンを押して画面内に初期サイズの矩形を作成し、位置設定する。その後ツールが消える。マウスカーソルにいちばん近い矩形角が該カーソルに取り付く。該矩形角の対角に新たなカーソルが現れ、該カーソルの位置は利き腕でない方の手で操作される。矩形の該両対角は同時に位置設定されることが可能であり、またスナップドラッギングを用いて即座に置かれることが可能である。マウスボタンがはなされると矩形が置かれてツールが再び現れる。この両手作図技法は他の図形の各位置に対しても使用されることが可能であり、該位置としては、線分の両端点、円の中心と円周上の点（円の平行移動とサイズ設定が同時に行なわれることが可能となる）、三角形の二角（三角形の平行移動、回転、およびサイズ設定が同時に行なわれることが可能となる）を含む。

２．０２　クリックスルーボタン

ほとんどのユーザーインターフェイスでは、ボタンが行う処理を記述するテキストは該ボタン自体のアクティブ領域内に置かれる。しかしオーバーレイ上においては、アクティブ領域を透明にしてその近辺にテキスト、アイコン、または処理を示す他の指標表示を伴わせるほうが好ましいことがしばしばである。これによりユーザーがボタン内に見える表示物に処理を行うことが可能となる。各アクティブ領域はクリックスルーボタンと呼ばれる。クリックスルーボタンは表示物のピックアップに用いることも可能である。

【００５８】図６は、削除、移動、およびコピー処理のためのクリックスルーボタン、および画面から表示物（楕円）を削除するための処理シーケンスを示す。ユーザーは削除ボタンが表示物グループの上になるようオーバーレイの位置設定を行い、この一方でこれら表示物のうちの一つにカーソルを置く。特定の処理の実行においてシステムは、マウスボタンがその上ではなされた時に処理されることとなる表示物をマウスボタンが押されている期間にわたり強調表示する。ユーザーがマウスボタンをはなすと選択された表示物が削除される。いくつかの表示物は削除ボタンに交差するが、ユーザーがマウスカーソルで示した表示物のみが実際に削除される。このことは処理対象表示物の厳密な特定を可能とする。またクリックスルーボタンはユーザーが処理と処理対象表示物を単一な両手動作で選択することを可能とする。ユーザーが前記と異なる処理を行おうとしていた場合は、別のクリックスルーボタンが使用されることが想定される。

【００５９】図７は、カラーパレットとして用いられるクリックスルーボタンのアレイ、および画面内の表示物（五角形）の色変更のための処理シーケンスを示す。この場合各ボタンは、右上部角に色を示す（各色は異なるパッチパターンで表される）三角形領域を有する矩形となっている。ユーザーはカラーパレットの所望の色をもつ部分を五角形上に置き、マウスでその上をクリックする。楕円も該ボタンの下となっているが、ユーザーがマウスカーソルで示す五角形のみがその色を変更する（ユーザーが表示物のない領域をボタンを通してクリックする場合、プログラムは該動作を無視するか、または該動作を省略した翻訳をすることが可能である）。従来のドローイングプログラムでは、ユーザーは（選択ツールを得るためにカーソルをツールパレットへ動かすような動作を必ず最初に行った後で）色が変更される表示物へカーソルを動かし、表示物の選択のため該表示物上をクリックし、そして所望の色の選択のためカーソルをカラーパレットまたはメニューへ動かすことが想定される。

【００６０】カラーパレットボタンは互いに境界を接するように示されているが、削除、移動、およびコピーボタンの場合のように互いに離れていてもよい。さらに、ボタン上の色領域は不透明部として示されているが、それは透明に近くてもよい。色領域の背面が透けて見える

場合、色領域はボタン領域全体をカバーすることが可能となる。表示物の輪郭色を変更するために、同様のクリックスルーボタンのアレイが提供されることが可能である。与えられるボタンの色は前述のように表示されることが可能であるが、色は三角形領域の外周のみに与えられるか、もしくはボタンの外周にわたり与えられるようにされる。

【００６１】図８は、ドキュメント内のテキストのスタイルを設定するための特性パレットとしてのクリックスルーボタンのアレイを示す。各スタイル（regular 、bold等々）はツール上にアクティブ領域を有している。この実施例ではボタン機能を記述するテキストがアクティブ領域内に位置している。該領域内に表示されているテキストの選択が該テキストのスタイルを変更する。この実施例ではユーザーはボールド(bold)ボタン内のテキストを選択しており、この結果選択されたテキストはボールドフェイスに変換される。プログラムがテキスト選択として認識する特定イベントは、該イベントが所望のボタンから始まっている限りさして重要でない。テキスト選択のメカニズムが所望のテキスト上のカーソルドラッグである場合、ユザーは選択のためにアクティブ領域内に開始位置を設定し、マウスボタンを押し、選択を完了させるためドラッグを行い、そしてマウスボタンをはなすことが想定される。マウスボタンがはなされる時点でカーソルはアクティブ領域の外となる傾向にあるが、これはどちらでもよい。

２．０３　クリップボード

クリップボードツールはその背面にある表示物から図形や特性をピックアップし、多くのアプリケーションで良く知られているコピーやペーストキイを表示化したもののように動作する。クリップボードは表示物全体、または色、線パターン、もしくはフォントのような特性をピックアップすることが可能である。クリップボードは単一または複数の表示物コピーを保持することが可能である。クリップボード上に捕捉された表示物または特性は、パレットツールの場合のようにその上をクリックすることで該クリップボードからコピーされることが可能である。クリップボードツールでピックアップされた表示物または形態は、ある意味でツールの一部となる。このことはユーザーがオーバーレイをカスタマイズすることを可能にする一般的特徴の一具体例にほかならない。該特徴については後に詳細な説明を行う。

【００６２】図９は、対称クリップボードの処理動作を示しており、該クリップボードはユーザーがクリックした図形をピックアップし、該図形を９０°ずつ回転させて得られるすべての図形を生成する。クリップボードを動かして再びクリックする結果、ユーザーは運ばれた該対称図形のコピーを作図領域上に落とす。クリップボードの左上角の小さい正方形のクリックは、新たな図形をクリップできるようクリップボードの内容を消去する。

【００６３】図１０は、表示物の図形的特性をピックアップしてこれを他の表示物に与える一対のツールを示す。図は楕円の色を矩形へ転写する特定シーケンスを示す。ユーザーは表示物の領域色に感受性をもつツールシートを通して楕円上をクリックする。領域色が図形から与えられて（または円形切片がその色ごとコピーされ）ツールシートの一部となる。該円形切片もその色がどこから来たかの指標として保持される。

【００６４】この後ユーザーはツールシートを動かし、特性アプリケータとして動作するツールシートの円形タブを矩形上に置く。ユーザーがマウスをクリックすると該矩形はツールシートから色を得る。二つめのツールシートは、別の表示物から二つめの充填色を得るために用いられることが可能であり、別のアプリケーションのためにいくつかの色を保存することを可能とする。

【００６５】図１１は、その背面にある表示物の形状をコピーし、該形状の選択された部分を再びアプリケーションに移すことをユーザーに可能とするツールの処理動作を示す。ユーザーがツールを通して表示物（この場合は曲線）をクリックした時点で、曲線またはそのコピーがツールの一部となる。後にユーザーが新たな図形を作成する時、該ツールは雲型定規のような使われかたをする。すなはち、ユーザーは新たな図形の近くにツールを置き、曲線のどの部分が新たな図形に付加されるかを特定するため曲線上の二点をクリックし（特定された部分は強調表示されてもよいし、色が変えられてもよい）、そして選択された部分がこれにいちばん近い新たな図形の端に付加される。

２．０４　クリックスルーボタンとビジュアルフィルタ

前述の各クリックスルーボタンでは、各クリックスルーボタンのアクティブ領域は完全に透明であり、ボタン背面の表示物をボタンがないかのように見せていた。しかし、適正に処理を行うために必要となる情報を強調してボタン背面の表示物を表すことが多くのアプリケーションに対し有用となることが想定される。このことは前述のビジュアルフィルタを用いて行うことが可能である。

【００６６】図１２を例にとると、ユーザーは多くの積み重ねられた矩形に対しており、中間にある矩形の左上部角を選択しようとしている。該角は最上部の矩形により覆われており、指示することが困難となっている。しかし頂点選択ボタンを有するツールが該角の所在を明らかにするワイヤーフレーム図（線状図）を表示し、これによりその選択を容易にする。

【００６７】ビジュアルフィルタとオーバーレイツールの組合せは特に有用なものとなり得る。従来のドローイングプログラムでは、表示物のワイヤーフレーム図を生成するためにユーザーは個別の命令を送出しなければならないことが想定される。該命令がいったん与えられると、処理対象表示物だけでなくすべての表示物がワイヤーフレーム図として作図されることが想定される。この

ことはユーザーの適正な処理対象表示物確認を助けるために重要となる該表示物の内容を失うことになる可能性がある。一方前記例ではユーザーは、局部表示処理、命令編集、および処理対象表示物これらすべてを単一な両手動作で一括指示する。

２．０５　オーバーレイとユーザー動作の合成

本発明のオーバーレイ技法はいづれの既存のユーザーインターフェイス技法とも組み合わせることが可能であり、また該組合せによりツールに興味深い特性を与えることが可能となる。

【００６８】このようなインターフェイス技法の一つとしてユーザー動作を利用するものがあり、該動作は通常一つまたはそれ以上のポインタ装置ストロークとなっている。一つのユーザー動作は通常一つまたはそれ以上の特徴的点（例えばストローク経路の開始および終了点、二つのストロークの交差点等）により特性づけられる。

【００６９】図１３は、オーバーレイを用いてパイメニューと単一ストローク動作を組み合わせるツールを使用する処理例を示す。該ツールは、多くの形態メニュー部分と原型表示物の保持領域に囲まれたアクティブ領域中心（表示領域）を与える。

【００７０】最初ツールの一部である原型表示物は、点線の外周と一つめの充填色を有する矩形である。ユーザーは実線の外周と二つめの充填色を有して画面内にある三角形上にツールの該中心領域を置く。三角形から充填色メニュー領域へのストローク（カーソルドラッグ）により、ユーザーは三角形の充填色を原型表示物に与えることを指示する。この時点で原型表示物の色が変更される。しかし該パイメニューは逆に原型表示物の特性を表示領域の表示物へ与えることにも用いられる。例えばユーザーが点線メニュー領域から三角形へストロークする場合、原型表示物の点線パターンが三角形へ与えられる。

【００７１】単一の特性だけでなく任意の特性集合または図形全体でさえも表示領域へ与えることを可能とする別のパイメニューが作られることも可能である。例えば原型表示物領域から表示領域へのストロークは原型表示物のすべての特性を指示された表示物へ与えてもよいし、また原型表示物自身を表示領域へコピーしてもよい。

【００７２】中心から外へ、または外から内へのストロークが可能なパイメニューはそれ自体で本発明に関する。しかしオーバーレイの内容物となっていないと意味をなさないことが想定される。片手動作パイメニューは、ストロークがはじまると該ストロークの始点上に中心を置いて現れる（文献Hopkins91）。従って外から内への初期ストロークの容易な方法はない。しかし本発明のパイメニューはオーバーレイ上にあることから、ストロークが始まる以前にパイメニューが現れてその内側へのストロークが可能である。ボタン内部および外部へストロークする考え方はパイメニューのような円形形態に限られない。いづれのボタン形状もこの機能を可能とする。

【００７３】図１４は、KurtenbachとBuxton（文献Kurtenbach91）による単一ストローク動作とオーバーレイの組み合わせを可能にする図形作成ツールを形成するための方法を示す。図は色のパレットから成るツールを示しているが、図７のような設計を用いることも可能である。ユーザーは作図画面内の所望の位置にカーソルを置き、所望の色がカーソル下となるようボタンを動かす。特定の色のボタン上からストロークを始めることで、ユーザーは該色の表示物を作成しようとしていることをシステムに告げる。ストロークの方向は作成される図形の形を決定し、ストローク長はそのサイズを決定する。ストロークが始まるとオーバーレイが消えて図形のパイメニューが現れ、ストローク方向と図形との対応をユーザーに示す。ストロークが完了すると新たな図形が画面に付加される。KurtenbachとBuxtonが述べているように、この処理の速度をさらに上げる多くの方法がある。例えばユーザーがストロークを瞬時に行った場合、パイメニューは現れる必要がない。

【００７４】ユーザーが単一の短い両手動作で新たな表示物の位置、色、サイズ、および形を特定することを可能にすることは注目すべきことである。従来のドローイングプログラムでは、ユーザーはまず矩形ツールの選択のためにカーソルをツールパレットへ動かし、カーソルを所望の開始位置へ戻し、所望の位置とサイズに矩形を作成し、そして所望の色選択のためにカーソルをカラーパレットまたはメニューへ動かすことが想定される。従来のカラーパレットは図７または図１４のクリックスルーボタンのアレイのように見えることが可能であるが、従来のパレットは通常、透明でないことが想定される。

２．０６　オーバーレイツールの画面内への取り付け

前述の各実施例では、画面上のオーバーレイの動きは画面の内容とは独立であった。しかし本発明は、画面内の一つまたはそれ以上の表示物と列をなすよう自身（またはオーバーレイシート全体）の位置を自動設定する有用なツールも提供する。後述の実施例では特定の表示物を画面上の各点に取り付けるために利き腕でない方の手が用いられることが可能であり、一方利き腕は他の処理を行う（または行う準備をする）ための自由な状態となる。

【００７５】図１５は、スナップドラッギング（文献Bier86）で用いられるようなルーラー・コンパス系の形成のための照準線を作成することに用いられるツールの例を示す。該ツールは共通点（円の中心）を異なる角度で通過する多くの照準線を有する。照準線のうちの一つ（例えば４５°の線）がソフトウェアカーソル（例えば図に示されているスナップドラッギングキャレット）の付近を通過する場合、該線はキャレットに取り付いて長

く伸び、ツールは選択された該照準線の傾きを表示する。この後ユーザーは該線を（例えばトラックボールボタンクリック、またはツール中心部の円上をマウスクリックすることで）その場所に固定する。この結果ユーザーはキャレットを照準線に取り付ける新たな処理を行うことが可能となる。図は、該照準線を指標に用いた新たな線分の作図を示している。

【００７６】前記スナップ技法は線以外の他の照準表示物に対しても用いることが可能である。図１６は、矩形の左下角にキャレットが置かれた図面上に位置設定された照準円パレットを示す。これら円のうちの一つ（例えば右の大きな円内の小さな円）の中心がキャレットの付近を通過する場合、該円は色を変えることで強調され、中心がキャレット先端へ正確に取り付く。この実施例ではパレット全体が取り付くが、単一の円のみが一時的にパレットの他の円から離れるようにすることも可能である。

【００７７】ユーザーはキャレット先端のような特定点以外の画面内の任意の点にツールを取り付けてもよい。図１７は、スナップドラッギングでサイズ設定中心または回転中心として用いられるアンカー（浮動固定点）表示物の例を示す。ここではユーザーはアンカーが矩形角近くになるまでオーバーレイを動かす。こうするとユーザーがそれ以上オーバーレイを動かさなくてもアンカーが矩形角に取り付く。ユーザーはオーバーレイを固定し、利き腕でアンカーの周りに矩形を回転させる。

【００７８】図１８は、回転、サイズ設定、および傾斜のいづれの処理も行えるように図１７のツールに汎用性をもたせたツールを示す。該ツールはアンカー位置の設定、処理の選択、および処理の実行を単一の両手動作で行うことを可能とする。すなはち、アンカー自身の周りに処理のパイメニューが置かれる。ユーザーが矩形角へアンカーを取り付けたところを再び仮定する。ユーザーは処理名、すなはち回転処理、上へカーソルを置きマウスボタンでクリックすることで処理を開始する。マウスボタンが押されると、システムはBierとStone の文献（文献Bier86）に記述されているスナップドラッギングのように表示物を回転させる。特に、アンカーを回転中心とした表示物の初期位置からの回転角度は、キャレットの動きを通して生じた角度と等しく保たれる。対話処理（この場合は回転）の期間中オーバーレイは消えていることが好ましい。

【００７９】画面表示物に取り付く前述の各ツールは、仮想ルーラー、コンパス、または分度器のような仮想作図ツールをユーザーに提供するオーバーレイの使用実施例ともなっている。現実のこれらのツールと同じく、利き腕でない方の手はツールの位置および方向を制御することが可能であり、一方利き腕は表示物または線の作成に用いられる。現実のツールを用いることと同様に、ツール付近に作られる表示物または角はツールの制約に影響される。この種のアプリケーションにオーバーレイを用いることの一つの利点は、パレットメニューの場合に見られる如くツールの制約が持ち込まれた後すぐさま容易にその影響をなくすことが可能になることである。

２．０７　オン／オフボタンとパレットメニューの合成

多くのシステムはトグルボタンを押すことでオンおよびオフできるような各状態を有している。さらにそのようなボタンはパレットメニュー上に置くことができ、この結果これらボタンを利き腕でない方の手によってカーソル付近に置くことによりカーソルのボタンへの移動量を低減でき、またユーザーの視線を作業領域からそらす必要をなくすことが可能となる。また、パレットメニューは大きなもの（全体が同時に画面上にある必要はなく、容易にスクロールオンおよびオフできるようにすることが可能である）になり得るため、より大きく目立つオン／オフボタン表示が用いられてもよい。

【００８０】図１９は、異なる傾きをもつ線として表示された一組のオン／オフボタンの例を示し、ここでユーザーはスナップドラッギング照準表示物のクラスを指定できるようになっている。画面内の矩形はその四つの角および中心をホットポイントとして明示している。ユーザーが特定の照準線（例えば垂直線）をクリックすると、該線は強調され、ホットポイントを有するすべての表示物は該クラスの照準線、すなはち垂直照準線、を最大に伸ばす。ツールはまた選択された照準線の傾きの数値指標を（右下のボックス内に）与えるよう示されている。ユーザーが一つ以上の傾きの異なる照準線を選択する場合、数値指標は表示されないことが想定される。

２．０８　仮想ガイドラインおよびグリッド

２．０４章ではオーバーレイとビジュアルフィルタの組合せについて説明した。これらの実施例でビジュアルフィルタは、表示物に対する処理を促進するよう通常とは異なる方法で画面表示物を提示した。しかしビジュアルフィルタはまた、通常ではまったく表示できない表示物を表示したり、そのような表示物に処理を施すことを可能にするために用いられることも可能である。例えばビジュアルフィルタはガイドラインやグリッドを局部的に表示することに用いられることが可能である。図２０は、それぞれ異なる種類のグリッド表示を行う三つのツールを示す。左の最初の二つのグリッドは間隔の異なる矩形グリッドである。最後のグリッドは六角形グリッドである。各グリッドはビジュアルフィルタが置かれた時のみ現れるが、オーバーレイが動いてもグリッド点が動かないようグリッドの座標系は画面に拘束される。従ってグリッド点をクリックしてオーバーレイを動かすことで、ユーザーはこれらグリッドを用いて画面を編集することが可能となる。ユーザーはグリッド点から線を引き始め、オーバーレイを上へ動かし、そしてグリッドを用いて線を引き終えたところである。従来のプログラムではグリッドの外観はそのグリッドをオンした時のみ見る

ことができた。しかし前記ツールを用いることで、ユーザーはいづれの与えられたツールについてもそれが与えるグリッドを使用前に見ることが可能となる。すべてのビジュアルフィルタツールはある程度まではこの特性を有している。

【００８１】図２１は、ユーザーにカスタマイズされたグリッドツールの作成および使用を可能とする方法を示す。カスタマイズされたグリッドツールは一組の選択された図形をピックアップし、ユーザーがこれをカスタマイズされたグリッドとして使用することを可能とする。該クリッドは他のグリッドと同様に表示物を正確に置くために使用されることが可能である。このツールは２．０３章でクリップボードとして説明したオーバーレイの特性を利用したものである。画面内の各線は、ユーザーが二列コラムレイアウトのテンプレートの仕様化のために（例えばエディタを用いて）作成したグリッドである。ユーザーは該グリッドをカスタマイズされたグリッドツールとして使用し、該ツール上で各グリッド線はツールの一部となる。しかしグリッドツールに関する前述の説明のように、各グリッド線はたとえツールが動かされても各位置を維持し、このためこれらは信頼性の高い基準点であり続ける。結果として、各グリッド線はグリッドツールが提示されている時のみ現れることとなる。図は、ユーザーが矩形を変形することを始め、最終的に矩形を左コラム下部へ取り付ける様子を示している。

【００８２】一つの可能な拡張は、いづれの画面内表示物もアプリケーションからオーバーレイ上へ移植を可能とすることである。このようにされた表示物は画面内表示物が取り付くよう引力が与えられることが想定され、これによりユーザーが自身のカスタマイズされたガイドラインおよびガイド曲線を作成することが可能となる。

２．０９　計測および定義ツール

前述の各ツールの作成は表示物の図形的特性を抽出することとなる。図２２は、幾何学的特性、すなはち座標、長さ、傾き、および角度等を計測するクリックスルーボタンツールの使用を示す。ユーザーが表示物の端点上をこのツールを通してクリックすると、該端点の座標がレポートされる。ユーザーがもう一回クリックする場合、システムは最初の点から二つめの点への長さおよび傾きをレポートする。ユーザーが三回クリックを行う場合、システムは最近クリックされた三点によってできる角度をレポートする。マウスが現在指示しているものに基づく情報を表示するツールはまた、ワードプロセッサに有用である。例えばツールはそれを通して選択された単語の定義を表示することが可能である。

２．１０　利き腕でない方の手による対象物指示

ほとんどのツールはオーバーレイの位置設定に利き腕でない方の手を用い、対象物指示には利き腕を用いるが、指示される表示物が大きい場合、指示に利き腕でない方の手を用いることも有意義なこととなる。例えばテキストドキュメントではパラグラフは通常大きいため、利き腕でない方の手で指示することが可能となる。図２３は、オーバーレイと共に動いて矢印先端下のパラグラフの隠れた構造を表すツールを示す。該ツールはパラグラフをプリンタ用にフォーマットするための印刷フォーマット名を常に表示する。この実施例ではボディーと言う名のフォーマットとなっているパラグラフを矢印は指示している。

２．１１　ユーザ動作翻訳ツール

オーバーレイの特に興味深い応用の一つは、ユーザーに完全な局部入力翻訳装置（またはプログラム）を与えることである。例えばドローイングプログラムは通常、マウス入力を選択、移動、線の作成等々の一連の命令として翻訳することが想定される（例えばRubineの文献のユーザー動作を用いた図形の編集（文献Rubine91）またはGoldbergとGoodisman の文献のユーザー動作を用いたテキストの編集（文献Goldberg91）参照）。ユーザ動作翻訳ツールを画面上に置くことにより、ユーザーはこれまでと異なる方法で同じエディタと対話することが想定される。例えば図２４において、ユーザーはユーザ動作翻訳ツール上にＸを描く。Ｘがツールにとって削除を意味する場合、Ｘ直下の表示物は削除されることが想定される。

【００８３】このようなユーザ動作翻訳装置（またはプログラム）は様々なアプリケーションで使用されることが可能である。例えばそれはマウスベースのワードプロセッサのユーザーインターフェイス層の最上位にユーザー動作インターフェイス層を形成する方法を与える。ペイントプログラムでは、ユーザ動作翻訳ツールが置かれていない時はマウスまたはペン動作をやめることが可能であり、また該ツールが置かれていれば命令編集処理が行われることが可能となる。さらにユーザ動作翻訳ツールにより与えられるインターフェイスは異なるアプリケーション間で共通化することが可能である。例えばオーバーレイがウィンドウシステムにより与えられる場合、同一のユーザ動作翻訳装置（またはプログラム）がドローイングプログラムからワードプロセッサへ移されることが可能であり、このことはユーザーが特定動作を複数の内容で使用することを可能とする。例えば図２４のＸは、ドローイングプログラムでは図形削除、ワードプロセッサではパラグラフ削除、またペイントプログラムでは領域削除に用いられることが可能である。

２．１２　局部命令翻訳装置（またはプログラム）とビジュアルフィルタの合成

局部命令翻訳装置（またはプログラム）を持たせる考え方はビジュアルフィルタと組み合わされることが可能である。例えば多くのドローイングプログラムはハンドルと呼ばれる小さいユーザーインターフェイス表示物を画面の表示物上に表示する。これらハンドルをカーソルで指示してマウスボタンを押すことで、ユーザーは表示物

の移動、サイズ設定、および変形処理等を行うことが可能となる。この考え方はオーバーレイと組み合わされて様々な異なる種類のハンドルを与えることが可能となる。例えば図２５において、ユーザーは画面内の三つの表示物から二つを選択したところであり、選択された表示物は小さな黒い正方形で強調される。変形ハンドルツールを置くことによりユーザーは一連の制御ハンドル（小さな白い正方形）のいずれをも見ることができ、また指示することが可能となる。中心ハンドル上でのクリックおよびドラッグは選択された表示物を移動させる。他のいづれかのハンドルのクリックおよびドラッグは選択された表示物を変形させる。

【００８４】一時的ツールの付加を行いアプリケーション表示物に関し置かれるビジュアルフィルタの有用性は、いくつかの該フィルタが単一オーバーレシート上で有効な場合特に明確に現れる。この場合ユーザーは一時的ツールから成る異なる集合を交互に利用することが可能となる。例えばあるツールの集合は前述のような移動および変形を可能とする編集ハンドルを与えてもよい。他のもう一つの集合は図形中心またはその図形のいづれかの角を中心とした回転を与えてもよい。これらツールが同時に与えられる場合、一時的ツールは受け入れがたい煩雑なものとなってしまうことが想定される。しかし前記二つのツールは、アプリケーション表示物の（位置、タイプ、数を含む）形態に依存した（位置、タイプ、数を含む）形態を有するツールの大きな多様性の利点をユーザーに与えるものであり、それゆえ該ツールは特別に調整されて該表示物に効果的に処理を行うものとなる。

２．１３　ログおよびデバッグツール

ツールはアプリケーションの通常処理に用いられるだけでなく、ユーザーがアプリケーションについてのヘルプ機能（コメント）を得たり、プログラマーがアプリケーションのデバッグを行ったりすることも可能とする。ツール境界内で行われる単一のユーザー動作またはマウス動作が、これからユーザーが行おうとする命令に関する情報の表示を該ツールに起こすようなツールがその例である。図２６は、マウスの絵を表示するツールの例を示す。カーソルがツール内にある間にユーザーがマウスボタンを押すと、ツール上に表示されたマウスアイコンがどのマウスボタンが現在押されているかを示す。このようなツールは、例えば対話ツールのビデオテープを作成する場合便利であることが想定される。この種のツールのより高度なバージョンは、実行されている命令の名前をも表示したり、該命令を実行しているソースコードを開示したりもすることが想定される。

２．１４　挿入点を用いる処理

クリックスルーボタンはオーバーレイによって作られることが可能な特に興味深い種類のボタンであるが、通常のボタンも有用である。図２７は、数字のキーパッドとして動作するボタンアレイの例を示す。該キーパッドはユーザーが作業しており、ペンやカーソルを用いて起動している領域付近に置かれることが可能であり、これによりいくつかの処理についてはキーボードを不要にする。マウスが数字をクリックするたびにオーバーレイは一文字の幅だけ右に動く。このキーパッドは計算機としても使用されることが可能であり、ユーザーが計算値をドキュメント中へ容易に挿入することを可能とする

２．１５　回転可能ツール

照準線ツールのような前述のツールのいくつかは、オーバーレイシートに対し平行移動が可能である。ツールがオーバーレイシートに対し回転やサイズ変更が可能なようにすることもできる。図２８は、タイプフェイスおよび／またはタイプテキスト選択のためのツールの例をしめす。任意の傾きにテキストを作成するため、ユーザーはツールを回転させることが可能である。この実施例ではユーザーがツールの計測領域（ツールの角）を通して画面上の二点をクリックする場合、ツールは該二点間を結ぶ直線の傾きに自身の方向をむける。その後ユーザーが該ツールからフォントを選択すると、該フォントの新たなテキストが前記の計測された傾斜で付加される。図に示された実施例は他の計測がなされるまで方向変更したままの状態を維持するが、ツールに一時的な方向変更をさせることもまた可能である。

２．１６　図形検索、ガイドラインおよび表示物作成の合成

前述のようにオーバーレイツールは、いくつかのタスクステップを単一な両手動作にまとめることが可能である。図２９は、ステップリダクションの効果を示す一実施例である図面ラベルツールを示す。該ツールは、規約ベースの図形検索（文献Kurlander92 ）、スナップドラッギング照準線、ビジュアルフィルタ、およびプッシュスルー表示物テンプレートをを組み合わせる。該ツールはすべての図面の矩形境界内の指定位置に図面ラベルを付加するために用いられる。該ツールが表示領域上を動かされると、表示領域内にあるすべての大きな矩形を見つけるために規約ベースの図形検索機能が用いられる。そのような各矩形に対し、該ツールは矩形の各端からある設定された距離をおいて照準（位置合わせ）線を描く。ユーザーはマウスを用いてツール表面上のテキストラベルのうちの一つを選択し、該ラベルをスナップドラッギングを用いて照準線に取り付ける。

２．１７　ウィンドウへのドキュメントロードのためのツール

一つまたはいくつかの表示物の図面への付加に加え、オーバーレイツールはディスクファイル全体をエディタウィンドウ、ドキュメントフレーム、または他の領域へロードするために用いられることも可能である。図３０はそのようなツールを示す。図の最初の部分はツールをしめしており、該ツールはドキュメントウィンドウの集合

上に置かれた多くのドキュメントアイコンを有する。ユーザーが一つのアイコンをクリックすると、対応するドキュメント（図の実施例では該ドキュメントはテキストと図形を有している）がカーソル背面のウィンドウ内に開かれる。図の実施例ではユーザーは選択されたウィンドウ内に置かれた一連の有効ファイルから家という名前のファイルを選択し、その後ファイル家の内容が選択されたウィンドウ内に表示される。これとは別のアプローチとして、ユーザーがツールを所望のウィンドウ付近に置きアイコンを該ウィンドウ内へドラッグする方法が考えられる。

３．０　オーバーレイのカスタマイズおよび使用

システムの設計者がどんなにユーザーの要求に整合しようとしても、ユーザーの見本のようなものは存在しない。従ってシステムは、ユーザーがオーバーレイツールとツールレイアウトを自身の基準に適合するようカスタマイズし、これから行おうとする特定タスクにオーバーレイを適合させることを許容することが予想される。本章ではユーザーがそうすることを可能にする多くの方法についての説明を行う。さらに、オーバーレイの使用に対し簡潔で一貫した取決めがある場合、オーバーレイはより効果的に使用されることが可能である。本章ではユーザーがより効果的にオーバーレイを使用することを可能とする多くの方法についての説明を行う。

３．０１　移動、コピー、および削除ツール

少なくともユーザーは、一種類以上のツールを作成してしまえばそれ以外のものは望まない傾向にあり、またツールを共通化して使えるようにそれをクラスターへ置くことを望む傾向にある。ユーザーがオーバーレイの形成に係わる能力を持たなければならない場合、ユーザーは少なくとも移動、コピー、および削除ツールの能力をオーバーレイシートから与えられなければならない。図３１は、この能力を与えるための、すなはちツール上でこれらの処理を行うハンドルを与える、一つの技法を示す。特定される実施例は、図９を用いて説明が行われた転写ツールである。図３１に示すように各ハンドルは所望の処理を与えるツール側部のアイコンである。ツールを動かすためにユーザーがハンドルをクリックしているところが示されている。実際には、ツール上のハンドルはおそらくもっと小さいはずであり、図に示されたものよりももっと大まかな表示のはずである。さらに、各ハンドルはツールの通常使用の間は表示されない（および効果を持たない）ように、またツール編集状態での処理にのみ寄与するように作られることが可能である。

３．０２　ツール編成

アプリケーションは自身のインターフェイス内に多くのツールを有する傾向にある。混乱を避けるため、これらツールとシートを編成してユーザーが直ちにいづれの所望のツールをも見つけることが可能なようにすることが必要である。一つのアプローチは、すべてのツールを単一シート上に置いてスクロールとズームでこれらを指示することである。各ツールは矩形面（タイル）上に編成させることが可能であり、このようにされた各矩形面を境界を接して結合させることで単一の大きなシートを形成することが可能となる。この後ユーザーは特定のツールを見つけるために所望の矩形面を画面上へスクロールし、そして所望のツールをその場へスクロールすることが想定される。クロールとズームを共に用いることで、ユーザーはかなり大きな矩形面内におけるツール指示が可能となることが想定される。加えて、トラックボールの動きとオーバーレイの動きとの間の対応は、トラックボールが素早く動かされた場合かなり大きい動きになり得ることから、小さいトラックボールの動きが必要となる。

【００８５】矩形面の数が非常に多くなる場合、該矩形面の編成に加えて階層構造をなす編成を使用することが可能となる。例えばユーザーは、それぞれが特定タスクに適用される多くのオーバーレイシートを有する仮想ボックスを作成してもよい。またこれとは別に、アレイとなった各矩形面は実際には矩形面の積み重ねであってもよい。ユーザーは現行画面に表示する矩形面をボタン上の繰り返しクリックまたはポップアップメニューの使用により選択することが可能である。ユーザーはまた（アレイよりもむしろ）一連のタイプ変化を行う単一の矩形面を使用することも可能である。

【００８６】本説明の原型ツールで用いられる技法は、単一のシートが異なるツール集合を異なる場面で表示することを可能とする。表示される集合はいくつかの方法で選択されることが可能である。（Apple ComputerのHyperCard（登録商標）の矢印の手法のように）ユーザーは集合内の特別なツールをクリックし、他の集合への転移を起こすことが可能である。加えて、マスター表示は有効な集合の内容表示を与え、これによりユーザーはいづれの集合へも転移が可能となる。異なる集合を同時に使用するためにユーザーは付加的シートを作成する。

３．０３　ツール合成－ビジュアルマクロ

クリックスルーツールとビジュアルフィルタはこれらをオーバーラップさせることで合成されることが可能であり、それゆえ従来処理を組み合わせて新たな処理を対話的に作成する能力をユーザーに与える。これは直覚的で強力なマクロ（複合）能力を与える。図３２は、充填色を（例えば赤に）変更するツールと線色を（例えば青に）変更するツールが合成されて充填色と線色の両方を変更するツールを形成する、ユーザーの立場から見た、様子を示す。該実施例で二つのツールは、それらのアクティブ領域が部分的にオーバーラップするよう一緒に運ばれる。ユーザーが該ツールのオーバーラップしていない部分をクリックする場合、結果として行われる処理は単一ツールにより通常与えられる処理のみとなる。一方ユーザーがオーバーラップ領域をクリックする場合、結

果として行われる処理は量ツールの処理の組合せとなる。

【００８７】ツールの合成は二つの方法で行うことが可能である。まず、二つのツールを一体となって動くよう同一オーバーレイシート上にオーバーラップすることが可能である。合成ツールが頻繁に使用される傾向にある場合、これは便利である。独立に動けるようツールを別々のオーバーレイシート上に置いてこれを合成することも可能であり、これらは特定の組合せが必要な場合に応じて一緒に運ばれる。

【００８８】オーバーラップされたツール（またはフィルタ）のスタックが入力（例えばマウスクリック）を受け取る場合、該入力イベントは該ツールの上から底へ通過させられる。各ツールは順に命令列を修正し、これはその後アセンブルされることとなる。例えばツールは現行命令列に付加的命令を接続してもよい。線色を青に変更するツールと合成されて充填色と線色の両方を変更するツールを形成する、ユーザーの観点から説明が行われた、前述の充填色を赤に変更するツールの実施例を考える。線色ツールが最上部にある場合、該ツール通過後の命令列はSetLineColor blue となることが想定され、また両ツール通過後の命令列はSetLineColor blue;SetFillColor redとなることが想定される。

３．０４　より複雑なマクロ

前述のツール合成はマクロ操作のための強力な技法である。しかしユーザーが数個以上のツールを合成したい場合、ツールが煩雑になって視覚直覚的利点が失われる危険がある。これをふまえてオーバーレイは合成ツール作成のための別の技法を与える。該技法はいくつかのプログラムで見られるマクロ記録機能に類似したものである。これは通常、ユーザーがシステムをマクロ作成のための状態に置き、その後所望の処理シーケンスを単一表示物に施すことを必要とすることが想定される。システムは該表示物に行われるすべての処理の合成ツールを自動作成することが想定される。

【００８９】前記合成ツールは多くの利点を有する。例えばまだ作成されていない表示物を含む多くの表示物にあるまとまった処理を行いたい場合、追加される表示物に対する処理は所望の個別処理から成る全体シーケンスを再び施すよりも単一ステップのほうが都合がよい。加えて、ユーザーは単一表示物または同時に選択された表示物集合の両方に対しあるまとまった処理を容易に施すことが可能となる。

３．０５　選択表示物の記憶

ユーザーが表示物の集合を選択し、各表示物に対し同時にまたは個別に処理を行う場合、いづれの場合も時に問題を生じる。処理を行った後、ユーザーは該処理を修正または取消たい場合がある。オーバーレイは、そのツールがどの表示物に適用されたかを記憶する能力を各ツールに持たせる技法を与えることが可能である。各表示物集合に処理を行った後に別の処理が必要であることに気がついた場合、ユーザーは各表示物を思い出したり明確な選択処理を行うことなしに該表示物集合を取り戻すことが可能となる。ツールがいちばん最近選択された図形集合しか記憶しない場合でもこのメカニズムは有用である。

３．０６　ツールの作成および変更

ツールを作成および変更するための技法のいくつかについてはすでに説明を行った。これらはオーバーレイを構成するための移動、コピー、削除、およびオーバーラップツールについての前述の規定と、表示物の形や特性をコピーして２．０３章で述べたような特別なクリップボードツールを作成するための技法とを含んでいる。本発明はユーザーがオーバーレイツールの作成および変更をさらに柔軟に行えることを意図している。例えば新たなオーバーレイツールの幾何学的構造物を作成するためにドローイングプログラムまたはワードプロセッサが用いられることが可能であり、その後埋め込みボタン構造（文献Bier90,Bier91a,Bier92 ）を用いて該構造物に特性が与えられる。

【００９０】さらに、オーバーレイの一つのシートを他のシートの編集に使用することも可能である。そのような環境ではグラフィカルエディタ環境での作図と同様の形式でオーバーレイシートが編集可能になることが想定され、また前述の様々なツールや技法がもり込まれることが可能となる。さらに、利き腕出ない方の手が位置変更だけでなくオーバレイシートのサイズ変更も可能とする場合、ユーザーは特別なアプリケーションのためにツールを大きくしたり小さくしたりすることが可能となる。

【００９１】本発明ではオーバーレイツール作成のために二つのツールキットを用いる。一つめはオブジェクト指向プログラミングを通してツールが作成される従来のツールキットである。二つめのツールキットは埋め込みボタン技法に基づくものであり、ユーザーはグラフィカルエディタを用いて新たなツールまたはツールの集合を作図し、その後これら図形に特性（例えばクリックスルーボタン機能等）を与える。該特性はユーザーがカスタマイズした言語で表される。

３．０７　ズームおよびスクロール

アプリケーション上でのオーバーレイシートの位置設定については、ユーザーにとって等価な二つの方法がある。ユーザーがオーバーレイを表示画面上で動かしている間、表示画面が固定され続けることが可能であり、またユーザーがオーバーレイ下の表示画面を動かしている間、オーバーレイが固定され続けることが可能である。ユーザーにこの種の両機能を与えることは有意義なこととなる。アプリケーションのスクロールはユーザーが該アプリケーションの表示画面外の部分を表示画面上ツールへ運ぶことを可能にし、広面積なアプリケーションの

操作を可能にする。アプリケーション上でのオーバーレイのスクロールはユーザーが表示画面外のツールを表示画面上のアプリケーション表示物へ運ぶことを可能にし、広面積なオーバーレイシートの使用を可能にする。表示体に対するオーバーレイの位置設定を言う場合は、平行移動だけでなく回転も含めて考えられなければならない。

【００９２】利き腕を使いマウスボタンを押してシート境界上をドラッグすることでオーバーレイシートの移動が可能となるが、利き腕でない方の手と二つめの入力装置（例えばトラックボール）の利用が好ましい。トラックボールボタンとこれらの機能の対応の一方法は次のようになる。左のボタンクリックはトラックボールとサムホイールをそれぞれ表示画面（アプリケーション）のスクロールとズーム用にする。右のボタンクリックはトラックボールとサムホイールをそれぞれオーバーレイのスクロールとズーム用にする。中間のボタンクリックは表示画面とオーバーレイを一体にしてスクロールとズームが行われることを可能とする。このような対応はユーザーによりカスタマイズされることが可能である。例えばユーザーがオーバーレイを必要ない時は消し、必要な時は再び現わすことを簡単に行いたい場合、トラックボールボタンの一つにこのオンおよびオフのトグル機能を持たせてもよい。さらに、複数のオーバーレイシートがある場合については、各シートが独立にスクロールおよびズームされることを可能とする付加的対応を与えることが望ましい。これは異なるシートに対して異なるトラックボールボタンの組合せをユーザーがクリックするようにすることで可能となる。

【００９３】これらの移動およびサイズ制御を用いることでユーザーはいづれのアプリケーション表示物上にもツール中心を置くことが可能となり、また所望の表示画面領域をカバーするようツールのサイズ変更を行うことが可能となる。いくつかの表示物にツールを適用する場合のシート動作を最小にするため大きなツールが用いられることが可能である。作業領域全体をカバーするよう引き伸ばされたツールはアプリケーション全体にわたり効果的に命令状態（モード）を設定する。前述のツールのいくつかについては、オーバーレイをある意味で固定できるようにすることが必要である。例えば２．０６章のスナップツールは、画面表示物が取りつくことができるよう照準線が固定されることを必要とする。（例えば中間のトラックボールボタンがクリックされて）オーバーレイシートが表示画面に固定される場合、これらツールをオーバーレイシートに対して固定することが一つの慣例になると想定される。オーバーレイのズームやスクロールと同様の効果を与えるものとしてツールレイアウトが考えられる。この場合あるツール集合がより大きく、また他のそれがより小さく描かれ、より大きく描かれた該集合がユーザーの制御下に入る。これはある意味でツールの魚眼表示に類似するものであり、ユーザーが多くの表示画面スペースを占めることなく多くのツールを見ていくつかのツールを選択することを可能とする。

３．０８　オーバーレイ移動のためのキーボード

前述の各実施例はトラックボールのような入力装置を利用しており、該入力装置はオーバーレ内の大きな動作および小さな動作の両方を一連の位置信号として容易に起動可能である。一方キーボードのようなオンオフ変化のみを知らせる入力装置から一連の位置信号を与えることもまた可能である。例えばコンピュータキーボードのあるまとまったキーがオーバーレイを一定方向に一定距離だけ動かすようにすることが可能である。これらのキーを一回またはそれ以上押すことでユーザーは与えられたワークピース上に異なる枠取り領域を置くことが可能となる。このアプローチを用いてオーバーレイ上のツールレイアウトを正常に動作するよう調整することが可能となる。例えば各ツールは標準アレーにレイアウトされることが可能であり、この場合の全体間隔はオーバーレイ移動キーのクリックでオーバーレイが移動する距離の倍数となる。

【００９４】これとは別に、キーボードキーは反転クラッチとして使用されることが可能であり、これによりオーバーレイが容易にマウスの動きと結合したり離れたりすることが可能となる。このキーが押されている場合オーバーレイはマウスカーソルと共に動く。このキーがはなされるとオーバーレイはその時の位置に止まり、マウスカーソルは独立に動く。

【００９５】オーバーレイのスクロールに関してのキーボード使用は、トラックボールやマウスで構成されていないコンピュータのユーザーに比較的容易なオーバーレイ使用を可能にする利点がある。

３．０９　状態ツール

両手使用のユーザーはマウスカーソルとツールの両方を表示画面上で動かすことで様々な表示物に繰り返して処理を行うことが可能であるが、これは多くの設定を必要とし、またオーバーレイとマウスカーソルの位置変更を逐次に行わなければならない片手使用のユーザーにとって都合のわるいこととなる傾向にある。この問題はオーバーレイ環境とより従来形態に近いインターフェイスとの間でスムースな転移を可能とすることで克服することが可能である。例えば３．０１章で述べたツールハンドルは、カーソルを該ツールに対応するツール状態に置くためのボタンを含むことが可能である。該状態期間中ユーザーはあたかも該ツールを通してクリックしているかのように繰り返して処理を行うことが可能となる。該状態が継続していることの指標としてカーソルは該ツールに似た外形を帯びることが可能である。

【００９６】クリックスルーツールが一時的な状態ツールになることを可能にすることは、一時的状態ツールを他のツールと共に動作させることを可能にするかどうか

の問題を提起する。一つのアプローチは、一時的な状態ツールがクリックスルーツールが合成されるのと同一の方法で他のツールと合成されることを可能とすることである。もう一つのアプローチは、一時的な状態ツールが素データに処理を行える唯一のツールとなるようシステムに要求することである。すなはち、ツールを該状態に置くことはこれ以外のツールの使用を排除することが想定される。そのような場合オーバーレイシートがボタンにより再び置かれてユーザーが該状態を出ることが可能である。そのような状態から出たり入ったりすることはユーザー動作を用いて行うことも可能である。例えばツール内のダブルクリックで該状態に入ってもよい。ダブルクリックを再び行うことで該状態から出ることが想定される。

３．１０　ツールのドラッグアウト

通常のツールの使用は、マウスカーソルをツール内へ動かすこと、マウスボタンの一つを押すこと、および行われるべき中間動作の後マウスボタンがはなされることとを含む。マウスボタンが押されることとはなされることの間には多くのことが考えられる。これらは、マウスがまったく動かないこと、マウスが動いてツール内に止まること、マウスがツールを離れること、およびマウスがツールを離れて再び戻ってくることとを含む。処理がツール内で始まる場合、マウスボタンがはなされるまで該処理が継続することが一つの慣例になり得る。該慣例はツールを比較的小さくすることを可能とし、一方ユーザーが処理をツール内で始めてその後該処理をツール外へ継続することを可能とする。

３．１１　クリックスルーツールおよびオーバーレイと従来ツールの合成

前述のように従来ツールはオーバーレイツールと共に、またはオーバーレイツール自体として使用されることが可能である。本発明はクリックスルーツールを伴うオーバーレイが既存のプログラムの強化機能として与えられることを意図する。従来のユーザーインターフェイスを利用する初期バージョンプログラムに親しんだユーザーは、強力で新たなユーザーインターフェイスを前にすると慎重に事を進めることが想定される。従ってユーザーは従来ツールのほとんどをマウスカーソルで可動な一連のパレットとしては保持するが、該ツールはほとんどオーバーレイには付加されないことが想定される。ユーザーがオーバーレイの使用および利き腕でない方の手の使用にさらに親しんで短時間で容易にツール設定を行うようになるに従い、ユーザーはより多くのツールをオーバーレイに付加することが可能になる。ユーザーは最初従来ツールのみを用いて一つまたはそれ以上のオーバーレイシートを作成し、いくつかのクリックスルーツールの試用を始め、その後にクリックスルーツールと従来ツールを同一シート上に混在させることが想定される。

【００９７】従来ツールとクリックスルーツールを合成することもまた可能である。例えば図７に示すようなクリックスルーカラーパレットは、一組の従来形態の作図ツールと合成されることが可能であり、該ツールの場合ユーザーはツール上をクリックしてカーソルを作図領域へ動かし、そしてカーソルをクリックまたはドラッグして図形を形成する。そのような合成は、所望の色のクリックスルーボタンを所望の従来の作図ツール上に重ね合わせることでユーザーが色と図形を単一ステップで選択することを可能とする。ユーザーはそれでもまだ従来方法でカーソルを動かし図形を形成しなければならない。これは図１４に示されるツールを特徴づけるツール動作の全体的効率化を与えないが、それでもユーザーは単一クリックで色と図形を選択可能になることで効率化を得る。状態ツールもまたクリックスルーボタンおよびその複合体と共に用いられることが可能であり、該ボタンは線幅、点線形態、およびこれらに類似の他の表示物の特性を特定し、該複合体は一つ以上の表示物特性の特定に用いられる。

【００９８】前記のような合成のもう一つの実施例は、ユーザーが最初に従来の作図ツールをクリックし、その後所望の一連の特性の特定のために一つまたはそれ以上のクリックスルーボタンを通して作図を進める場合である。この方法では従来の作図ツールのクリックにより表示物の外形があらかじめ選択されているため、ユーザーは単一ステップで描かれるべき表示物の特性および位置の選択が可能である。

４．０　特定の実施

本章ではオーバーレイツールおよびビジュアルフィルタを含むシースルーインターフェイスの一般的実施について説明を行う。ソフトウェアは、Sun Microsystems SPARCステーションおよびこの他のコンピュータ上のSunOS UNIX（登録商標）と互換性のあるオペレーティングシステム上で走るセダープログラム言語および環境（文献Swinehart86）内のMulti-Device Multi-User Multi-Editor（MMM）構造体（文献Bier91b）内で実行される。MMM内に組み込まれたガーゴイルグラフィクスエディタ（文献Pier88）がインターフェイステストのための複合アプリケーションとして動作する。ユーザー入力装置は利き腕用に標準的マウスを、また利き腕でない方の手のためにMicroSpeed FastTRAP（登録商標）トラックボールを含む。該トラックボールは三つのボタンおよびサムホイールを含み、これらはインターフェイスに付加的パラメータを与える。

【００９９】本章では三つのオーバーレイサブシステムについて説明を行う。一つは二つのポインタ装置からの同時入力を処理して複数の同時変化の後の表示画面の更新を行うものであり、もう一つは位置指定イベントがツールおよびビジュアルフィルタを通過する時に該イベントの変更を行うものであり、そしてもう一つは図形出力が各ツールおよびビジュアルフィルタから送出される時

に該出力の変更を行うものである。

４．０１　複数装置入力

シースルーインターフェイスは、次に述べるようなMMMの特徴に依存している。MMM はマウスやトラックボールのような複数の入力装置からイベントを受け取り、どの装置がどのイベントを発生したのかを追跡し、そしてすべてのイベントを単一の待ち行列上に置く。MMM は各イベントを待ち行列から順に解除してどのアプリケーションに該イベントが送出されるべきかを決定する。各MMMアプリケーションは階層構造に置かれており、該構造は各アプリケーションが表示画面上でどのように入れ子にされるかを示す。各イベントはルートアプリケーションへ送られ、該アプリケーションはイベントをその子供（下位）のアプリケーションへ送ることが可能であり、下位のアプリケーションはイベントを階層ツリー上の更に下位へ順に送ることが可能である。マウスイベントは一般に入れ子の最深部のアプリケーションへ送られ、該アプリケーションの表示画面領域はマウス座標を有している。しかしユーザーが特定なアプリケーション内で表示物のドラッグやサイズ変更を行っている場合、ドラッグやサイズ変更が完了するまですべてのマウス座標は該アプリケーションへ行く。キーボードイベントは現在選択されているアプリケーションへ行く。オーバーレイをサポートするため、トラックボール入力を処理するためのMMM の規約が変更された。オーバーレイが可動な場合トラックボールおよびサムホイールイベントは最上位のアプリケーションへ行き、該アプリケーションは各イベントをそれぞれシートの移動およびサイズ変更命令として翻訳する。シートが可動でない場合トラックボールおよびサムホイールイベントは選択されたアプリケーションへ送られ、該アプリケーションは各イベントをそれぞれ該アプリケーションのスクロールおよびズーム命令として翻訳する。

【０１００】図３３は、ユーザー入力ルーチンのフローチャートを示し、該ルーチンは入力イベントに応答してとられる適正な動作を決定する。オーバーレイを有するシステムが入力を受信する場合、システムはイベントがオーバーレイの移動、カーソルの移動、オーバーレイの起動、または通常の方法による従来アプリケーションへの該イベントの送出のどれを意図するかを決定し、その後該決定に従って動作しなければならない。ルーチンは最初に入力がオーバーレイ移動装置からかどうかをテストし、もしそうなら該装置の動作機能に従いオーバーレイを動かす。もしそうでない場合、ルーチンは入力がオーバーレイサイズ変更装置からかどうかをテストし、もしそうなら該装置の動作機能に従いオーバーレイのサイズを変更する。もしそうでない場合、ルーチンは入力がカーソル移動装置からかどうかをテストし、もしそうなら該装置の動作機能に従いカーソルを動かす。次に、それが適正なものであればイベントはルートアプリケーション（これは例えばウィンドウマネージャーであってもよい）へ送られ、該アプリケーションはイベントが次にどこへ送出されるかを決定する。

【０１０１】オーバーレイ移動装置という言葉は、オーバーレイを動かすものとして現在指定された装置を意味する。装置テーブルと呼ばれるソフトウェアデータ構造体がある装置をオーバーレイ移動装置として現在指定していれば、特定の物理的装置は特定な時点でオーバーレイ移動装置となる。この指定はユーザーによりいつでも変更可能である。同じことがオーバーレイサイズ変更装置という言葉にもあてはまる。

４．０２　ツールおよびビジュアルフィルタによる入力のフィルタ処理

MMM 入力イベントは通常、ルート（根）アプリケーションからリーフ（葉）アプリケーションへ向かって規則正しく移動する。しかし本発明を実行するシステムは各アプリケーションにわたり共通な多くのオーバーレイシートを有してもよい。オーバーレイをサポートするため、各入力イベントは階層ツリーの下から上へ送り返されなければならない。図３４の（Ａ）は、アプリケーション＃１、＃２、＃３、＃３Ａ、＃３Ｂおよび＃４、およびオーバーレイシート＃１および＃２の間のアプリケーション階層構造の例を示す。図３４の（Ｂ）は、これらのアプリケーションウィンドウとオーバーレイシートが該階層構造に対しどのように画面上に現れるかを示す。

【０１０２】アプリケーション＃１はルートアプリケーションであり、そのウィンドウはこれに関連する他のすべてのアプリケーションウィンドウを有している。この実施では該アプリケーションは最上位のMMM 矩形エディタとなっていることが想定され、該アプリケーションはウィンドウシステムとして動作する。この階層ツリーの上から二段めの左から右への順序は表示画面上の上から下へのアプリケーションの順序を指定しており、従ってアプリケーション＃４が最上部のアプリケーションとなる。オーバーレイシート＃１はアプリケーション＃３と＃４の間に置かれており、一方シート＃２はアプリケーション＃４の上に置かれる。アプリケーション＃３Ａおよび＃３Ｂはアプリケーション＃３のウィンドウ内に含まれるように示されているが、これらは別の方法で＃３のウィンドウと関係づけられてもよい。

【０１０３】入力イベントは最初にオーバーレイへ送られ、ユーザーがツールまたはビジュアルフィルタと対話しようとしているかどうかが決定される。もしそうなら該イベントはオーバーレイにより変更される。いづれの場合でもイベントはルートアプリケーションへ戻され、該アプリケーションは該イベント自体を受け入れるか、さもなくば階層ツリーのさらに右方向に現れる下位のアプリケーション上へ該イベントを送る。図３４の（Ｂ）に示すように、各オーバーレイシートのアクティブ領域内およびアプリケーション＃３上にカーソルが置かれる

場合の実施を考える。マウスボタンを押すことのような起動イベントをユーザーが与える場合、該イベントは最初にシート＃２を通り、次にシート＃１を通り、そしてカーソル座標を有するアプリケーションであるアプリケーション＃３B へ送られることが想定される。

【０１０４】MMM イベントを表すデータ構造は三つの方法で変更されてオーバーレイをサポートする。一つめの方法として、既に通過したアプリケーションツリーの部分を表す注釈（belowChildおよびbelowTool と呼ばれるイベントフィールド）がイベントに付加される。これによりルートアプリケーションが二回以上シートへイベントを送ることが避けられる。二つめの方法として、所望のアプリケーションへ到達したときに翻訳される命令列がイベントに付加される。例えばカラーパレットクリックスルーボタンは各マウスクリックイベントに指定色で修飾されたSetFillColor命令を付加する。最後の方法として、ツールがビジュアルフィルタを有する場合、イベントが所望の表示物に対し適正に指示されるよう該イベントのマウス座標が修正され、ビジュアルフィルタを通して該表示物がカーソル下に現れる。

【０１０５】図３４の（C）は、前記イベントが適正なアプリケーションへ送られる様子を示す。特に、イベントが適正な装置座標を伴い適正な順序で各シートおよびアプリケーション層へ送られるようにするため、イベントはアプリケーション階層構造を上下に移動する。

４．０３　イベント送出ルーチン

前記４．０２章では、入力イベントが適正なオーバーレイシートを通って所望のアプリケーションへ送られる方法について説明を行った。本章以降では、フローチャートと疑似符号を参照し、いくつかの重要なルーチンについてのイベント処理のより詳細な説明を行う。該ルーチンは、ユーザー入力ルーチン、イベントーアプリケーションルーチン、イベントーオーバーレイルーチン、イベントーツールルーチン、翻訳および実行ルーチン、および合成ルーチンである。

【０１０６】図３４の（A）から図３４の（C）に示される前述のようなオーバーレイを通してのアプリケーションへのイベント送出は、次の三つのルーチンを使用している。

【０１０７】イベントーアプリケーションルーチン；このルーチンはイベントが標準的アプリケーションへ送られる場合に実行される。

【０１０８】イベントーオーバーレイルーチン；このルーチンはイベントがオーバーレイへ送られる場合に使用される。

【０１０９】イベントーツールルーチン；このルーチンはイベントがオーバーレイ上の特定ツールへ送られる場合に実行される。

これら各ルーチンは他の一つまたはそれ以上のルーチンを呼び出してもよい（例えばイベントが標準的アプリケーションからオーバーレイへ、またはオーバーレイから標準的アプリケーションへ送られる場合であり、このようなことは前述の実施例で何回か起こる）。

【０１１０】与えられたイベントE は、階層ツリー内の各アプリケーションおよびシートへ入るごとにフィールドbelowChildを持ち、該フィールドはアレイ（または他の多値データ構造）を有する。各通過段階での論理値belowChild A は、A の子供（下位）であるアプリケーションまたはシートへのポインタであり、またイベントがこれまでに入った最後がA であることを示している。同様なフィールドbelowTool は各通過段階でbelowTool 0 を持ち、これはイベントがこれまでに入った最後である 0 のポインタとなる。

【０１１１】図３５は、イベントーアプリケーションルーチンのフローチャートを示す。該ルーチンは新たなイベントが発生するたびごとにいくつかの変数が初期化されることを前提としている。例えばbelowChild A は、アプリケーションA の子供およびA のすべての子供の最上位として定義される仮想アプリケーションに対し初期化される。

【０１１２】図３６は、イベントーオーバーレイルーチンのフローチャートを示す。入力イベントがオーバーレイにより受信される場合、オーバーレイはそれが有する各枠取り領域、すなはちツールのうちどれが該イベントを処理するのかを決定する。いくつかの場合には数個のツールが順にイベントを処理する。ツールがクリックスルーツールである場合、該ツールは命令リストと呼ばれるデータ構造を算出し、これが該ツール背面の各ツール（もしあるなら）またはオーバーレイ全面の背後にある各アプリケーションへ（オーバーレイの親プログラムを通して）送られる。該ルーチンは新たなイベントが発生するたびごとにいくつかの変数が初期化されることを前提としている。例えばbelowTool 0 は、0 の背面にある最上部として定義される仮想ツールに対し初期化される。オーバーレイが最初に作成される場合、それはユーザー動作状態になく、それが有するいづれのツールもユーザー動作処理ツールにならない。

【０１１３】図３７は、イベントーツールルーチンのフローチャートを示す。イベントE が特定なオーバーレイツールT により受信される場合、T はE のタイプからどの動作A が行われるべきかを決定する。T が従来の（非クリックスルー）ツールである場合、T は即座に動作を行う。そうでない場合T はそのオーバーレイ背面のアプリケーションへ該イベントを送るか、もしくは直接または他のツールを通してそのオーバーレイまたは他のオーバーレイへ該イベントを送る。E が他のツールによりすでに処理されている場合、T はすでにイベントE と関係を持った他のいづれかの動作とA とを合成する必要がある場合がある。

４．０４　翻訳および実行ルーチン外観

41

与えられる動作A は一般に、適当なプログラム言語で書かれた任意の符号体であってもよい。該言語はC やPascalのようないづれの汎用プログラム言語であってもよいが、もしそれが翻訳された簡潔な言語である場合はオーバーレイツールのユーザーカスタマイズ化をより良くサポートする。そのような言語にはApple社のHyperTalk、Jhon Ousterhout によるTcl （文献Ousterhout90）、またはInterleaf （文献English90 ）や埋め込みボタン構造（文献Bier90,Bier91a,Bier92 ）で用いられているようなLISPの小規模バージョンがある。各アプリケーションが高密度処理状態で走るUNIXのような環境中でのオーバーレイツール使用をサポートするため、A はX ウィンドウシステムにより与えられるような中間処理メッセージとして表されることが可能である。オーバーレイツールの一般的実行は、小規模なLISP類似言語を用いたセダープログラム環境（文献Swinehart86 ）内、および動作表現のためにX ウィンドウメッセージを用いたUNIX/Xウィンドウ内で行われる。

42

＊【０１１４】与えられたソフトウェアプログラムP が動作A を受信した場合、該プログラムは自身の文中で該動作を翻訳しなければならない。いくつかのプログラムにより処理が可能な動作をA が記述している場合、P はA を自身の言語に翻訳した後自身（P ）のデータ構造上でA の処理を行わなければならない。符号体を特定な文（スコープとも呼ばれる）へ翻訳することは良く知られたコンピュータサイエンス技法である。A が正常に翻訳されるために特定なプログラムP はA で使用されるすべての変数に対し結合子（論理値）をあたえなければならない。後述の各実施例は前述の一般的実行における該技法の特定な使用を示しており、該技法はアプリケーションと独立なツールに適用される。

４．０５　汎用プログラムを用いた動作

例えばマウスカーソル背面にあるいづれかの矩形のサイズを二倍にすることを意図する動作A は、次の疑似符号に似た符号を有することが想定される。

【０１１５】

```
FOREEACHSHAPE,s,in PICTURE do
    ifISRECTANGLE s  and INCLUDES s, cursorPoint
        then SCALE s, 2.0
endloop
```

該動作を翻訳するためP は前記の各単語に対して自身の定義を与えなければならない。P は、図面内の各図形にわたる繰り返しを指示するFOREEACHSHAPE に対し自身の処理動作を、変数 PICTUREに対し自身の論理値を、図形が矩形かどうかをテストするISRECTANGLE に対し自身のルーチンを、また図形のサイズ変更を行うSCALE に対し自身の処理手続きを与えなければならない。加えてP は、前述の各オーバーレイルーチンによりP に送られる現行のカーソル位置を変数cursorPoint が有していることを明確にしなければならない。

【０１１６】オーバーレイ動作を簡潔なものとするため、セダーにおけるオーバーレイの一般的実行は前述の実施例に示されるような汎用プログラムをサポートしない。その代わりとして一般的実行は各動作を命令リストに置き換える。この場合各命令は一連の論理値で修飾される命令符号となる。該論理値はオーバーレイで算出され、アプリケーションでは算出されない。

４．０６　一つまたはそれ以上の命令を用いた動作

例えばカーソルから四分の一インチの範囲内でカーソル点<23,37> 付近にある最上部の表示物をP が選択することを規定する単一命令は、次のようになることが想定される。

【０１１７】 (SelectObject<23,37>0.25 )

LISP言語のように命令全体が括弧でくくられ、命令の始まりと終わりがどこであるかを容易に知らせる。表示物を選択して充填色を赤に、輪郭色を青にする次に示す動作のように、各動作はいくつかの命令を有してもよい。

【０１１８】

((SelectObject<23,37>0.25)(SetFillColor red)(SetLineColor blue))

この場合一対の付加的括弧が動作全体の始まりと終わりを表す。これは前述のフローチャートにおいてL として引用されている命令リストの一実施例となっている。

４．０７　命令の翻訳

このような動作表現は一般的原型ツールで特に容易に翻訳される。これはオーバーレイがすでに実行されているソフトウェアプログラムはみなこの形式で表現された命令を翻訳するためである。従って前述のプログラム依存のない命令リストを特定なプログラムP により翻訳可能な命令リストに翻訳するために、各汎用命令をP が理解できる一つまたはそれ以上の命令に翻訳することが必要となる。例えばP はSelectObject命令を有さずにSelect命令をその所望機能と共に有してもよい。Selectはその二つめのアーギュメントをインチ形式の代わりにピクセル形式とすることが想定される（72ピクセル／インチの場合0.25インチ=18 ピクセル）。同様にP はSetFillColor命令を有さずにSetAreaColor命令を有してもよい。PがSetLineColor命令を有する場合、命令リストの適切な翻訳の一つは次のようになる。

【０１１９】((Select<23,37>18)(SetAreaColor red)(SetLineColor blue))

命令列の長さが常に保持されるとは限らない。例えば該動作は(SetFillColorAtPoint cursorPoint red) 命令を有してもよく、これはカーソルにいちばん近い表示物の色を赤に変える。特定プログラムP はこのような処理を単一命令として与える必要はない。この場合単一命令(S

etFillColorAtPoint cursorPoint red) は(Select cusorPoint)(SetAreaColor red) の二つの命令に翻訳されてもよい。

【０１２０】P がそのような命令リストに対する翻訳をまだ実行しておらず、所望の機能を与える一連の処理手続きを与える場合、各動作は直接処理呼び出しに翻訳されることが可能となる。例えば前記動作は次のようなセダープログラム言語の三つの処理呼び出しに翻訳されることが想定される。

【０１２１】SelectShape picture, 23,27 ,18 ;
shape''GetSelectedShape picture ;
SetAreaColor shape,red ;
SetLineColor shape,blue ;

４．０８　反転データフロー

これまでに述べた動作は、命令およびデータをツールからアプリケーションへ伝達するものであった。命令によりアプリケーションからツールへデータを戻すこともまた可能である。このことは命令リスト中にいくつかの空白な記憶領域を持たせることで可能となる。例えばコピーおよびペーストツールは次のようにして該動作を与えることが可能である。

【０１２２】
((Select<23,37>18)(GetSelected fillInValue))
この場合fillInValue は後の論理値記録のためのラベルfvのポインタである。

【０１２３】図３８の（A）および３８の（B）は、命令により要求されたデータをアプリケーションが戻す前後での該記録を示す。該記録はフィールドvalue を含み、該フィールドにアプリケーションにより戻されるデータに対するポインタが置かれる。二つめのフィールドready?はフィールドvalue に論理値が与えられるまで論理値false を有する。最後のフィールドcondVar はセダープログラム言語により与えられる状態変数と呼ばれる同期表記を有する。状態変数は現在これを待っている各プログラムのリストを含んでいる。該変数が有効になると、該変数に論理値を与えるアプリケーションが現在該変数を待っている各プログラムにこれを知らせる。

【０１２４】この実施例ではアプリケーションP が座標<23,37> で図形を選択した後フィールドvalue にポインタを記憶する。P がT と異なる処理過程を走っている場合、T はP がポインタを記憶するまで待たなければ（状態を保留しなければ）ならない。これはT がそのポインタをfvに主張しており、いったんvalue が受入状態になったことが知らされると該ポインタがfvのフィールドvalue に到達することが可能になるためである。

４．０９　合成ルーチン

二つまたはそれ以上のオーバーレイが（図３４の（B）のオーバーレイ＃１および＃２のように）オーバーラップされる場合、各イベントE は前面から背面の順に（オーバーレイ＃２の後オーバーレイ＃１）すべてのオーバーレイにより処理された後最終的に受け取られるべきアプリケーション（図３４の（C）のアプリケーション＃３B）へ送られる。結果的に、動作A はいくつかのまたはすべてのオーバーレイからの寄与を含む処理を記述することが可能である。そのような動作は各オーバーレイにより独自に決定されることが想定される各動作の合成としてとらえることができる。現在の実行では各オーバーレイO はそれぞれの動作とイベントがその時点までに累積した動作とを合成する役目を負っている。新たな動作は様々な方法のうちの一つを用いて既存の動作から算出されることが可能であり、該方法は追加、先頭追加、削除、命令変更、アーギュメント変更、または座標変更等を含む。これら各方法について以下に説明を行う。

４．０９．０１　追加

その動作を追加するツールは既存の命令リストの終端に該動作の命令リストを単純に付加する。例えばオーバーレイO がその座標<x,y> がO のツールT で指定されるイベントE を受信する場合、およびツールT が輪郭色を変更するツールである場合を考える。なにものにも依らず、イベントE に対するT の対応は次のような動作を発生することであると想定される。

【０１２５】
((SelectShape<x,y>)(SetLineColor blue))
しかしE が充填色を赤にすることを規定するツールをすでに通過していた場合、次のような動作が行われたことが想定される。

【０１２６】((SelectShape<x,y>)(SetFillColor red))
T が追加ツールである場合、T はその命令リストを既存の命令リストへ追加して次のようなより長い動作を形成する。

【０１２７】((SelectShape<x,y>)(SetFillColor red)(SelectShape<x,y>)(SetLineColor blue))

４．０９．０２　先頭追加

先頭追加は新たな命令リストが既存の命令リストの先頭に追加されることを除き追加に類似している。命令の順序づけが関係する場合、先頭追加は追加と異なる結果を与える場合がある。例えばイベントE が図形を45° 回転させる既存の命令(RotateShape45) を伴って入ってくる場合、およびツールT が図形をy 方向はそのままでx 方向に二倍する命令(ScaleShape21)を通常発生する場合を考えると、最終的命令は((ScaleShape21)(RotateShape45)) となることが想定される。これは((Rotate45)(ScaleShape21))とは異なる効果を有する。

４．０９．０３　削除

ツールT はイベントから一つまたはそれ以上の命令を取り除くことが可能である。例えばT は表示画面を編集するすべての命令を削除することでその直下の図面が編集されないようにすることを可能とする。T がイベント((SelectShape<x,y>)(SetLineColor blue)) を受信した場合、背後の図面を修正していたはずの(SetLineColor

blue) 命令は取り除くが、その時図面内で選択される表示物の変更のみを行う(SelectShape<x,y>)命令は残しておくことが想定される。この結果生じる命令は(SelectShape<x,y>)となることが想定される。

４．０９．０４　命令変更

ツールT は受信された動作で使用されているいくつかまたはすべての命令表記を変更することが可能である。例えばT はSetFillColorへのすべての呼び出しをSetFancyFillColor へのすべての呼び出しに、またSetLineColorへのすべての呼び出しをSetFancyLineColor への呼び出しに変更することが可能である。このことはユーザーがオーバーレイ0 の前面にあるオーバーレイ上の親しんだツールを使用しつつ新たな装飾的命令を試行して作りだすことを可能とする。そのようなツールが命令

((SelectShape<x,y>)(SetFillColor red)
(SelectShape<x,y>)(SetLineColor blue))

を受信する場合、該ツールは命令

((SelectShape<x,y>)(SetFancyFillColor red)
(SelectShape<x,y>)(SetFancyLineColor blue))

を作成することが想定される。

４．０９．０５　アーギュメント変更

アーギュメント変更の場合、T は命令の命令表記の後に規定される論理値を変更する。例えばT はその前のツールにより受信されたいづれかの色を変更してそれらをより鮮明にすることが想定される。そのようなツールが命令(SetLineColor blue) を受信する場合、該ツールは命令(SetLineColor vivid-blue) を作成することが想定される。

４．０９．０６　座標変更

座標変更はアーギュメント変更の特別な場合である。この場合ツールT は命令内に規定された座標<x,y> を変更する。T が0 の背面にある図面の表示を変更するビジュアルフィルタを有する場合、このことは特に有用になる。例えばT が直下の図面を拡大する場合、T の前面にあるツールからの座標は該ツールからの命令がT を通して実際に現れる表示物に適用されるよう縮小変換値を有さなければならない。従ってT がその背面の図面を二倍に拡大する場合、T は該座標を0.5倍にしなければならない。受信された命令が(SelectShape<x,y>)である場合、TはSelectShape(0.5*<x,y>)を作成する。

４．０９．０７　マーク座標について

イベントーオーバーレイルーチンは、翻訳のためにマークされた命令リスト内の座標<x,y> を認識する。一般にカーソル座標はこのようにマークされる。該マークは合成ルーチンにより保持され、この結果入力命令がマーク座標を有する場合には合成により作成される該座標に対応した命令内座標もマークされる。

４．１０　基本的出力処理

背面にあるアプリケーションプログラムへの入力イベントの送出に加え、オーバーレイはこれらアプリケーションプログラムの表示の修正も行う。特にほとんどのツールは該ツールのアクティブ領域の境界を示すアイコンを表示する。加えてツールはビジュアルフィルタを有していてもよく、該フィルタはアプリケーションの表示のフィルタ処理、外形、色、サイズ、およびその他の図形的特性の修正を行う。本章以降ではオーバーレイメカニズムがどのようにビジュアルフィルタの動作を制御するかについての説明を行う。

【０１２８】MMM 出力は通常、リーフアプリケーションからその上位へ向かって合成される。ビジュアルフィルタをサポートするため、通常の表示画面リフレッシュ処理が拡張されて情報が階層ツリー内を自由に流れることが可能となる。例えば図３４B のツールが一つまたはそれ以上のビジュアルフィルタを有し、またこれらビジュアルフィルタのいづれかがグラフィカルエディタ上に設定されている場合、各ビジュアルフィルタは自身を描くために（階層構造ではビジュアルフィルタの同胞である）該グラフィカルエディタの内容を検定しなければならない。

４．１０．０１　変更領域の算出

オーバーレイにより与えられる表示画面リフレッシュルーチンのいくつかの部分は入力処理のルーチンに類似しているが、データはこれと逆方向へ送られる。図３４の（C ）に示されたように、オーバーレイルーチンは入力イベントが各オーバーレイを前面から背面の順に通過して目的のアプリケーションへ送られることを保証しなければならない。アプリケーション＃3B の内容が変更される場合に表示画面のどの部分が書き換えられなければならないかを算出するため、アプリケーション＃3B の変更部分の情報が図３９に示される経路に沿って送られる。これは図３４の（C ）の経路の逆経路となっている。

【０１２９】変更領域情報はこのツリー内のノードからノードへ送られるため、変更領域はこれを受信する各ノードの座標系に変換され、各ノードにより容易に理解される。加えて各オーバーレイノードは、その上に見えている現時点の表示フィルタに基づく変更領域を変更する可能性がある。例えば与えられたオーバーレイ0 が拡大フィルタを有するツールT を含む場合、0 は表示画面領域のサイズを増大させることが想定され、このことはアプリケーション＃3B 自体の変更により影響を受ける。0 は変更領域データ構造のサイズを適正に増大させ、変更されたデータ構造を連鎖内の次のノードへ送る。

４．１０．０２　表示画面更新

ビジュアルフィルタが無い場合、アプリケーションおよびオーバーレイ両者を表す新たなスクリーンイメージの作成は、階層構造内の各ノードを層順を優先して上層から順に、また同一層では下位から順に背面から前面へ描くことで行われる（すなはち各アプリケーションはその子供が描かれる前および後の両方で描かれる）。図３９

の実施例では、各ノードは次のような順序で描かれることが想定される。アプリケーション＃１開始、アプリケーション＃２開始および終了、アプリケーション＃３開始、アプリケーション＃３A 開始および終了、アプリケーション＃３B 開始および終了、アプリケーション＃３終了、オーバーレイ＃１開始および終了、アプリケーション＃４開始および終了、オーバーレイ＃２開始および終了、アプリケーション＃１終了。

【０１３０】しかしオーバーレイがひとつまたはそれ以上のビジュアルフィルタを有する場合、そのような各フィルタはビジュアルフィルタの特徴である変換動作を用いて階層構造の部分が描き直されることを必要とすることが想定される。描き直される階層構造の部分は、オーバーレイの親および該オーバーレイの背面にある該オーバーレイのすべての同胞を含む。例えばオーバーレイ＃１がビジュアルフィルタを含む場合、アプリケーション＃２、アプリケーション＃３A 、アプリケーション＃３B 、アプリケーション＃３、およびアプリケーション＃１のノードが（この順に）描き直される。図４０は、このツリー断片を示す。

５．０　利点

本章ではオーバーレイの様々な実施例の多くの利点を要約する。これら利点のいくつかは前述の様々なツールに関連付けて明確に説明を行ったが、これらの利点をまとめておくことは有用である。

５．０１　タスクのより高速な実行

いくつかのステップを単一ステップにまとめることで、オーバーレイツールは時間を節約することが可能である。前述のようにクリックスルーボタンは命令選択と処理対象選択を単一なユーザー動作にまとめる。ビジュアルフィルタの使用を加えることで、これらボタンはまた処理対象領域内にカスタマイズされた表示を作成し、この場合該表示を明確にオンおよびオフするためのキーストロークはまったく必要ない。加えてユーザーの視線およびカーソルの両方が作業領域を離れることがない。このことはユーザーがタスクに集中することを持続させ、またユーザーが作業領域外に置かれたパレットと作業領域の間を繰り返しカーソル移動することを必要とするシステムに比較して時間の節約となる。

５．０２　時間的状態の減少

オーバーレイを用いることで、処理を与えるツール上にカーソルがある場合にはいつでも処理が可能となる。本質的にオーバーレイは空間的命令状態を与える。（例えばツール上のラベルにより）ユーザーが容易に現行状態を見ることができ、また（例えばカーソルをツールの外へ動かして）容易にその状態からでる様子を見ることができるため、空間的状態はしばしば時間的状態よりも望ましい。

５．０３　より良好なグラフィカルフィードバック、エラーの減少

時間的状態を用いるシステムは頻繁にユーザーにフィードバックを与え、現行命令の正常な完了を助ける。例えばグラフィカルエディタのテキスト編集状態では、エディタはすべての編集可能なテキスト列を強調表示することが可能である。オーバーレイツールはこれと同じタイプのフィードバックを与えることが可能であるが、それはツールの境界内に限られる。実際、スクリーン上でいくつかのツールを同時に用いることでいくつかの異なるフィードバックが各ツール内に与えられることが可能となる。加えて、ツールをエディタの全体サイズに拡大することで、ユーザーは時間的状態を用いることで可能となるフィードバックと同じフィードバックを可能にする。いくつかのツールが同時に見える場合、各ツール内のフィードバックは二つの役割を行う。それはユーザーがツールを適正に使用することを助け、またユーザーが適正なツールを選択することを助ける。

【０１３１】加えて、ツール内でのビジュアルフィルタの使用は、従来与えられることのなかったフィードバックの一種を与える。これは表示画面の一部を修正して表示し、一方表示画面の残りの領域はその内容の標準的表示を行う。ビジュアルフィルタは隠れた頂点のような隠れた情報を表示することが可能である。加えて、それは与えられた処理に対する適正な処理対象となる表示物の特定を助けることが可能である。例えば表示物の境界色を変更するツールは境界の外形のみを表示することが想定される。そのようなフィードバックは、充填色の変更のような前記と異なる目的のためのツールを誤って使用したユーザーにとっては不都合になることが想定される。

５．０４　容易なカスタマイズ化

表示画面スペースを占領してしまうようなツールを有するアプリケーションでは、ユーザーカスタマイズ化の効果は限られたものとなる。ユーザーが新たなボタン、スライダー、およびこの他の対話装置を作成できたとしても、これらを置くスペースがほとんどない。しかし本質的にオーバーレイは制限を受けないスペースを与え、該ツールが作業領域全体から取り払われるだけでなく、それらは表示画面領域外へ置かれて必要なときに表示画面上へスクロールされることも可能となる。加えて、オーバーレイツールとビジュアルフィルタはオーバーラップすることで合成されることが可能なため、これらはユーザーが個人仕様のマクロを構築する容易な方法を与える。

５．０５　学習時間の軽減

オーバーレイツールはいくつかのタスクステップを単一にまとめることが可能なため、ユーザーがこれらステップを個々に学習する必要が無くなる。例えば表示物を選択してそれに色を与えることを学習する代わりに、ユーザーはカラークリックスルーボタンの使用を学習する。同様に、初心者が自然とエキスパートになる。熟練した

ユーザーは各ツールが置かれている場所、および効果的なツールの合成方法を知っており、またツールを短時間に所望の位置へ動かすことをすでに学習している。各ツールの空間的配置およびツールの合成を学習すること、および共通して使用される動作をこれがユーザーの運動神経の一部となるまで繰り返すことで初心者はエキスパートになっていく。これらのタスクは自覚的努力をほとんど必要としない。

５．０６　アプリケーションに依存しないツール

いくつかのユーザー命令は多くのアプリケーションで共通なため、それらはカットおよびペーストキーのようにキーボード内に組み込まれる。ユーザーがそのようなキーの一つを押す場合、ウィンドウマネージャーは該キーストロークをユーザーの現行アプリケーションへ送る。オーバーレイはこのようなアプリケーションに依存しない命令が置かれる新たな基体を提示する。特にオーバーレイシートがアプリケーション間を平行移動できる場合、オーバーレイの各ツールは、その適用が意味を持つときはいつでもその背面にあるいかなるアプリケーションに対しても現行アプリケーションを識別する必要なく適用されることが可能である。例えば色を変更するクリックスルーボタンは、異なるエディタ内の各表示物の色を変更することが可能である。

５．０７　片手または両手による使用

多くのグラフィカルユーザーインターフェイスはユーザーが両手を連係させて使用することを必要とする。例えばユーザーはマウスを同時に使用する一方でコントロールまたはシフトキーを押し止めてもよい。オーバーレイは両手を用いてより素早く使用されることが可能であるが、片手によっても使用されることが可能であり、この場合オーバーレイとカーソルの位置設定が逐次的に行われる。オーバーレイは、片手が使えないユーザーまたは片手を電話やコーヒーカップを持つことのようなタスクに使用している器用なユーザーによっても操作されることが可能である。

５．０８　異なった表示サイズ

現在のコンピュータはポケットサイズから壁掛けサイズの多くの異なるサイズのディスプレイを備えている。オーバーレイはこのようなサイズ範囲にわたりインターフェイスを行う。オーバーレイツールはスクロールおよびズームを行うため、固定ツールパレットのためのスペースがほとんど無いかまたはまったく無い小さなディスプレイ上で使用されることが可能である。加えて、非常に大きなディスプレイ上ではユーザーは各ツールを表示画面のいづれの部分へも動かすことが可能であり、これは固定メニュー領域にアクセスするためにディスプレイ上を横切るような動きを不必要なものとする。

６．０　結論

本発明はユーザーインターフェイスの新たな形態を提供する。該インターフェイスは、アクティブなツール定義領域を有する透明なオーバーレイの表示に基づくシースルーインターフェイスである。該シースルーインターフェイスは、時間的状態よりもむしろ空間的状態に基づくユーザーインターフェイスのための新たな設計領域を提示し、また両手を使用した対話のための自然な情報媒体を提供する。該インターフェイスは可動であり、またアプリケーション領域上を覆うため、固定的表示画面領域をまったく持たず、また広範囲なディスプレイサイズに容易に適用が可能である。オーバーレイツールはオーバーレイを動かすことだけで選択されて作業領域へ運ばれるため、ユーザーの視線が作業領域に継続して注がれることが可能となる。動作および表示が空間的に定義されるため、ユーザーは全体的内容物を変更することなく作業をおこなうことが可能である。

【０１３２】さらに本発明のオーバーレイは、ビジュアルフィルタと共に用いられる場合、スプレッドシート内の同一物、グラフィカルエディタ内の表示物集合、または建築モデル内の水道管のようなアプリケーションの隠れた場所に対する処理を可能とする。ユーザーはこの隠れた場所を表示して編集することが可能である。この場合、固定的表示画面領域および命令の記憶はまったく必要ない。

【０１３３】加えて、シースルーインターフェイスは開放型ソフトウェアアーキテクチャをサポートする新たな範例を提供する。オーバーレイツールは単一のアプリケーションウィンドウに拘束されることなくアプリケーション間にわたり動かされることが可能であるため、該ツールはいくつかのアプリケーションの共通機能に対するインターフェイスを提供し、またより多くのアプリケーションに共通機能を提供することが可能である。

【０１３４】さらに、オーバーレイシートは非常に多くのツールを有することが可能なため、ユーザーが自身のカスタマイズしたツールおよびツール集合を作成することが可能な新たな可動基体を提供する。実際、該シートはユーザーにユーザーインターフェイスエディタを提供し、該エディタはユーザーが既存のツールを移動およびコピーすること、ツールをオーバーラップさせてマクロを合成すること、およびスナップツールを付加して新たな構成にすることを可能にする。あるオーバーレイシートが他のシートを編集するために用いられることさえも現実に可能である。

【０１３５】前述の説明は、本発明の詳細な実施例に関する完結した説明となっているが、様々な変更、別の構成、および前述と等価なものが用いられてもよい。例えばオーバーレイツールに関する前述の説明はアプリケーションを図形編集に限っているが、前述の多くのツールは本質的にいづれの表示画面ベースのアプリケーションにおいても使用されることが可能であり、該アプリケーションはスプレッドシート、テキストエディタ、マルチメディアエディタ、ペイントプログラム、ソリッドモデ

ラ、サーキットエディタ、サイエンティフィックビジュアライザ、およびミーティングサポートツールを含む。オーバーレイはビデオおよびコンピュータグラフィクスを含む画像メディア上で使用されることが可能である。これはかなり強制的なアプリケーション領域である。動いているメディアを見ているユーザーにとって視線をワークピースから画面側のメニューへ動かすことは、重要なイベントを見失う可能性があるためかなり不都合なこととなるからである。画像メディアの内容物中ではメディア自身がボタンを起動するイベントを与えてもよい。例えばアニメーションフレームの表示物の色をオレンジにする処理を正常に行うため、ユーザーは該表示物が現れる場所にボタンを押し止めておくことが想定される。新たなフレームが描かれるたびに該ボタンはフレーム内の該場所で該表示物へ自動的に適用されることが想定される。加えて、前述の説明ではアプリケーションプログラムに関連したオーバーレイツールの使用が強調されたが、オーバーレイはウィンドウシステム内において各ウィンドウを置いたり、ウィンドウ境界色、キーボードマクロおよびこれに類似するもののようなウィンドウパラメータの設定に使用されることが可能である。

【０１３６】従って前述の説明は請求項で定義される本発明の範囲を制限するように捉えられるべきではない。

７．０　文献

Bier86 Eric A. Bier and Maureen C. Stone. Snap-dragging. In Proceedings ofSiggrapf ` 86(Dallas, Augest), Computer Graphics, Vol. 20, No. 4, ACM, 1986, pages233-240.

Bier88 Eric A. Bier. Snap-Dragging:Interactive Geometric Design in Two and Three Dimensions. Report No. UCB/CSD 88/416, April28, 1988, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720. または Xerox PARC technical report EDL-89-2.

Bier90 Eric A. Bier and Aaron Goodisman. Documents as user interfaces. InR. Furuta(ed). EP90, Proceedings of the International Conferencs on Electronic Publishing, Document Manipulation and Typography, Cambridge University Press, 1990, pages249-262.

Bier91a Eric A. Bier. Embedded Buttons:documents as user interfaces. In Proceedings of the ACM SIGGRAPH Symposium on User Interface Software andTechnology(South Carolina, November), ACM, 1991, pages45-53.

Bier91b Eric A. Bier and Steve Freeman. MMM:a user interface architecture for shared editors on a single screen. In Proceedings of the ACM SIGGRAPH Symposium on User Interface Software and Technology(Hilton Head, SC, November 11-13), ACM, 1991, pages79-86.

Bier92 Eric A. Bier. Embedded Buttons:Supporting Buttons in Documents. ACM Transactions on Information Systems, Vol. 10, No. 4, October 1992, pages381-407.

English90 Paul M. English, Ethan S. Jacobson, Robert A. Morris, Kimbo B. Mundy, Stephen D. Pelletier, Thomas A. Polucci, and H. David Scarbro. An extensible, object-oriented system for active documents. In R. Furuta(ed). EP90, Proceedings of the International Conferencs on Electronic Publishing, DocumentManipulation and Typography, Cambridge University Press, 1990, pages263-276

Goldberg91 David Goldberg and Aaron Goodisman. Stylus user interfaces for manipulating text In Proceedings of the ACM Symposium on User Interface Software and Technology(UIST ` 91, Hilton Head South Carolina, November), 1991, pages127-135.

Hopkins91 Don Hopkins. The design and implementation of pie menus. Dr. Dobb's Journal. Vol. 16, No. 12, December 1991, pages16-26.

kurlandder92 David Kurlander and Steven Feiner. Interactive constraint-based search and replace. In Proceedings of CHI ` 92(Monterey, California, May), Human Factors in Computing Systems, ACM, New York, 1992, pages609-618.

Kurtenbach91 Gordon Kurtenbachand Wiiliam Buxton. Isses in combining marking and direct manipulation techniques. In Proceedings of the ACM Symposium on User Interface Software and Technology(UIST ` 91, South California, November), ACM, 1991, pages137-144.

Ousterhout90 J. K. Ousterhout. Tcl:An embbedable command language. In winter USENIX Conference Proceedings, 1990.

Pier88 Ken Pier, Eric A. Bier, and Maureen C. Stone. An Introduction to Gargoyle:An Interactive Illustration Tool. Proceedings of the InternationalConferencs on Electronic Publishing, Document Manipulation and Typography(Nice, France, April). Cambridge University Press, 1988, pages223-238

Rubine91 Dean Rubine. Specifying gestures by example. In Proceedings ofACM SIGGRAPH ` 91, Computer Graphics, Vol. 25, No. 4, July 1991, pages329-337.

Swinehart86 Daniel C. Swinehart, Polle T. Zellweger, Richard J. Beach, RobertB. Hagmann. A Structural View of the Cedar Programming Environment. ACM Transactions on Programming Languages and Systems, Vol. 8, No. 4, 1986, pages419-490. またはXerox PARC Technical Report CSL-86-1.

【０１３７】

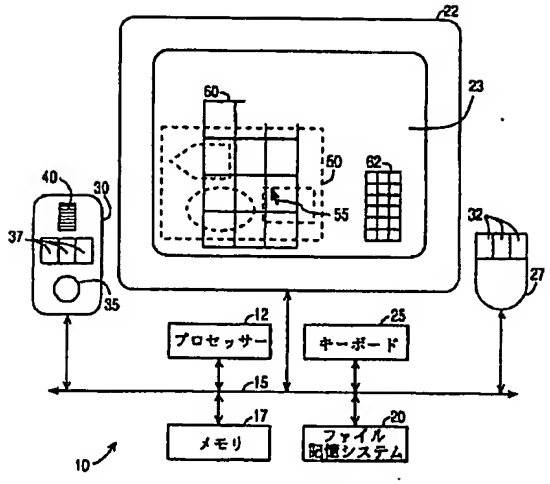
【発明の効果】本発明によりユーザーがより少ない動作で多くの一般的タスクを行うことが可能となり、この結果ユーザーの生産性が高められる。

【図面の簡単な説明】

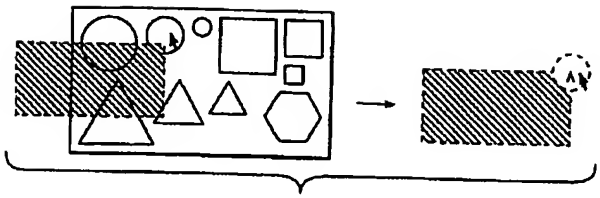
【図１】本発明の実施例であるコンピュータシステムの

ブロック図を示す。
【図２】プログラムおよびオーバーレイの素データを可視表示体に変換する方法を示す。
【図３】オーバーレイへの入力信号を処理呼び出しに変換する方法を示す。
【図４】作図ツールを示す。
【図５】作図ツールの特別な使用例を示す。
【図６】削除、移動、およびコピーツールを示す。
【図７】カラーパレットツールを示す。
【図８】テキストスタイルパレットツールを示す。
【図９】対称クリップボードツールを示す。
【図１０】表示物特性を転写するツールを示す。
【図１１】図形特性を転写するツールを示す。
【図１２】頂点選択ツールを示す。
【図１３】ユーザー動作を利用する特性選択ツールを示す。
【図１４】ユーザー動作を利用する色および作図ツールを示す。
【図１５】照準（位置合わせ）線ツールを示す。
【図１６】表示物に取り付く図形付加ツールを示す。
【図１７】回転ツールを示す。
【図１８】回転、サイズ設定、および傾斜ツールを示す。
【図１９】照準（位置合わせ）表示物を起動するツールを示す。
【図２０】グリッドツールを示す。
【図２１】カスタマイズしたグリッドツールを示す。
【図２２】幾何計測ツールを示す。
【図２３】テキストフォーマット表示ツールを示す。
【図２４】ユーザー動作翻訳ツールを示す。
【図２５】制御ハンドルツールを示す。
【図２６】デバッグツールを示す。
【図２７】数字キーパッドツールを示す。
【図２８】テキスト作成およびテキスト回転ツールを示す。
【図２９】ラベル付加ツールを示す。
【図３０】ドキュメントをウィンドウへロードするツールを示す。
【図３１】ハンドルを用いてツールを移動、コピー、および削除するツールを示す。
【図３２】ツールが合成されて新たなツールが作成される例を示す。
【図３３】特定な実行のためのユーザー入力ルーチンのフローチャートを示す。
【図３４】（A）はアプリケーションの階層構造を、（B）はアプリケーションの階層構造の画面表を、（C）はアプリケーションの階層構造のイベント送出順序をそれぞれ示す。
【図３５】特定な実施のためのイベントーアプリケーションルーチンのフローチャートを示す。
【図３６】特定な実施のためのイベントーオーバーレイルーチンのフローチャートを示す。
【図３７】特定な実施のためのイベントーツールルーチンのフローチャートを示す　。
【図３８】（A）は（アプリケーションがデータを戻す前の）データに対する要求を含む命令を、（B）は（アプリケーションがデータを戻した後の）データに対する要求を含む命令をそれぞれ示す。
【図３９】表示画面更新のためのイベント送出順序を示す。
【図４０】表示画面更新のための階層構造部分を示す。
【符号の説明】
１０　本発明の実施例であるコンピュータシステム
１２　プロセッサー
１５　バスシステム
１７　メモリ
２０　ファイル記憶システム
２２　ディスプレイ装置
２３　表示画面
２５　キーボード
２７　マウス
３０　トラックボール
３２　マウスボタン
３５　ボール
３７　トラックボールボタン
４０　サムホイール
５０　ドローイングプログラムのアプリケーションウィンドウ
５２　ワードプロセッサのアプリケーションウィンドウ
５５　カーソル
６０　オーバーレイツール
６２　パレット
７０　プログラム素データ
７２　翻訳装置（プログラム）
７３　プログラム画像データ構造
７５　オーバーレイ素データ
７７　翻訳装置（プログラム）
８０　オーバーレイ画像データ構造
８２　結合ノード
８３　ディスプレイイメージ
８５　オーバーレイ
８７　アプリケーション＃１
８８　アプリケーション＃２
９２　ウィンドウマネージャー
９３　翻訳装置（プログラム）＃１
９５　命令解析装置（プログラム）＃１
９７　翻訳装置（プログラム）＃２
９８　命令解析装置（プログラム）＃２

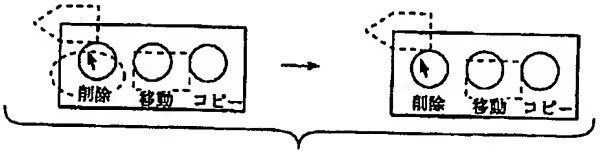
【図1】



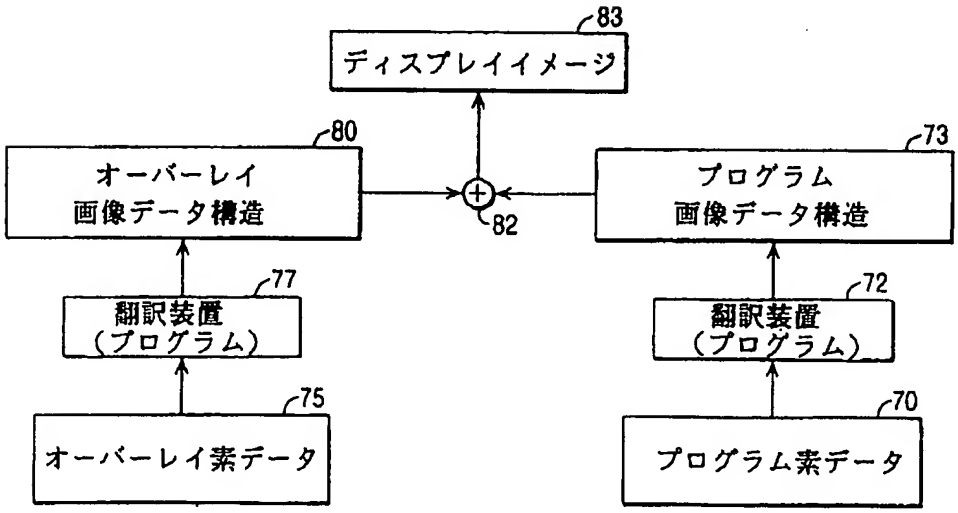

【図4】



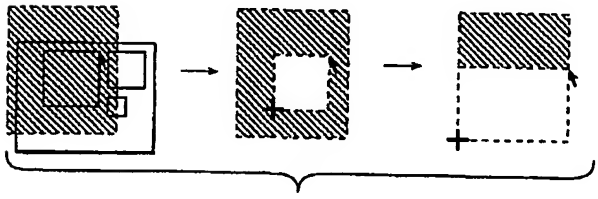
【図6】



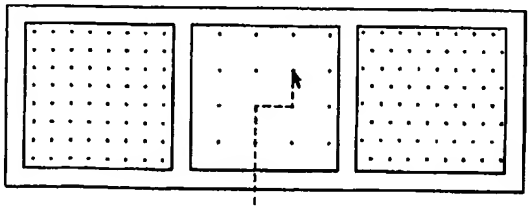

【図2】



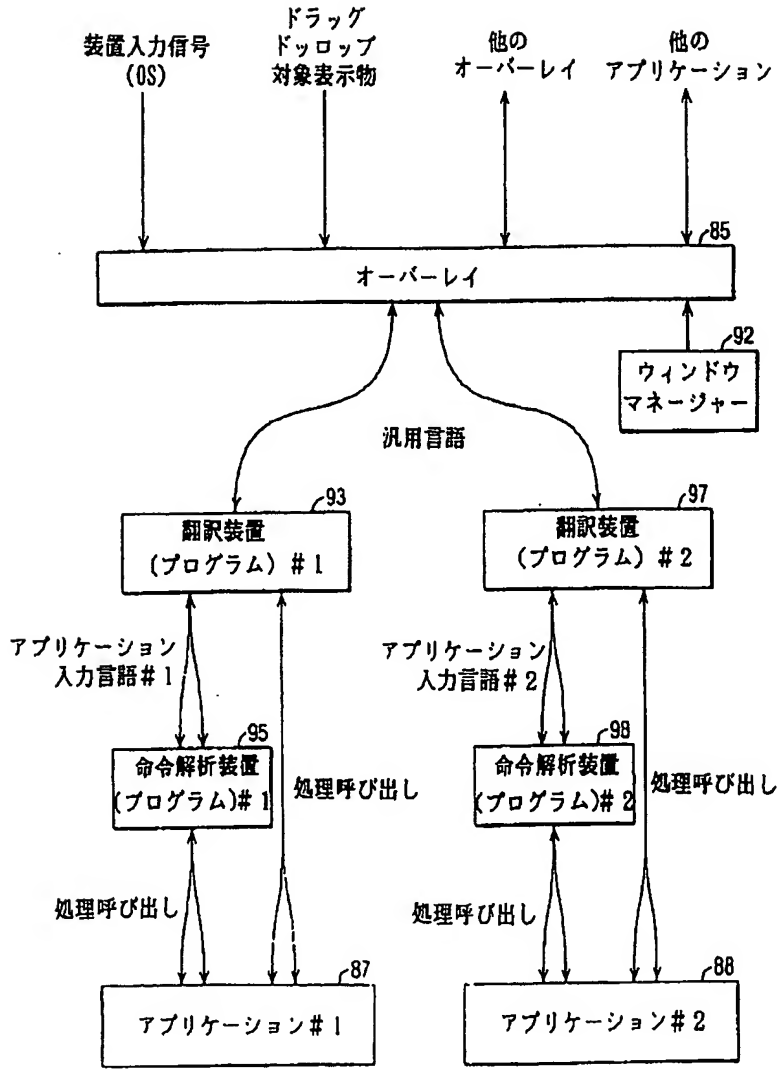

【図5】



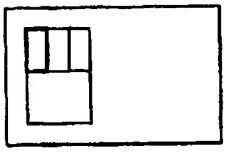
【図20】



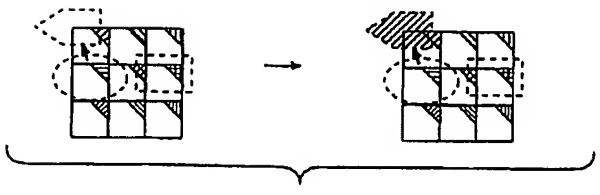
【図3】


装置入力信号
(OS)
ドラッグ
ドッロップ
対象表示物
他の
オーバーレイ
他の
アプリケーション
85
オーバーレイ
92
ウィンドウ
マネージャー
汎用言語
93
翻訳装置
(プログラム) #1
97
翻訳装置
(プログラム) #2
アプリケーション
入力言語#1
アプリケーション
入力言語#2
95
命令解析装置
(プログラム)#1
98
命令解析装置
(プログラム)#2
処理呼び出し
処理呼び出し
処理呼び出し
処理呼び出し
87
アプリケーション#1
88
アプリケーション#2

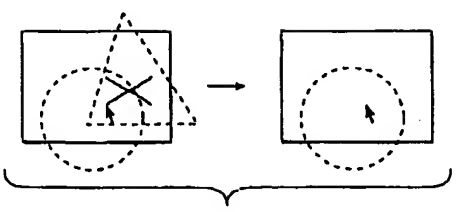
【図26】



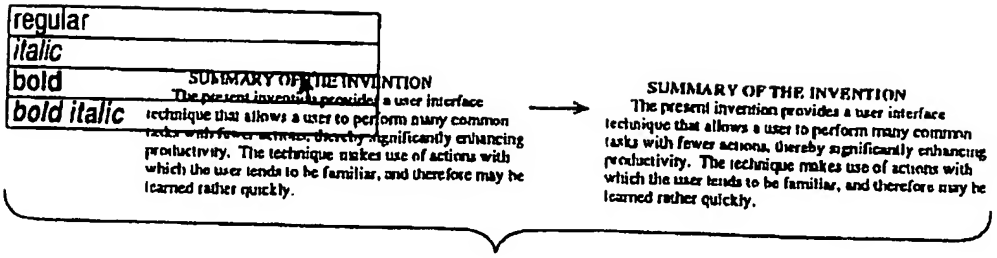
【図7】



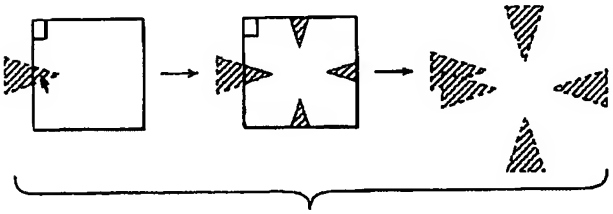
【図24】



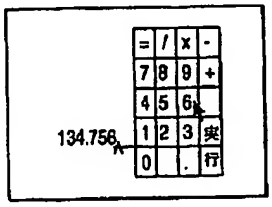
【図8】



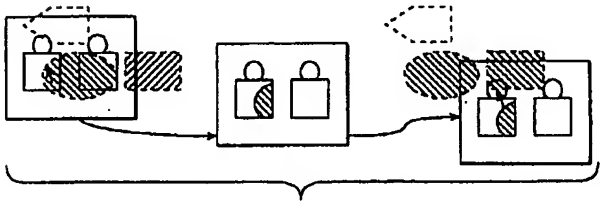

【図9】



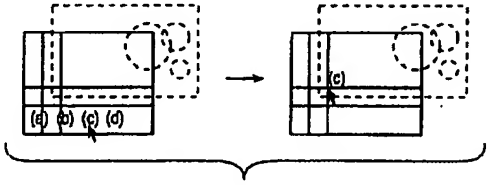
【図27】



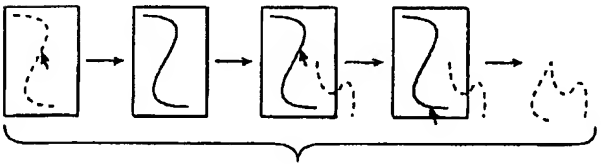

【図10】



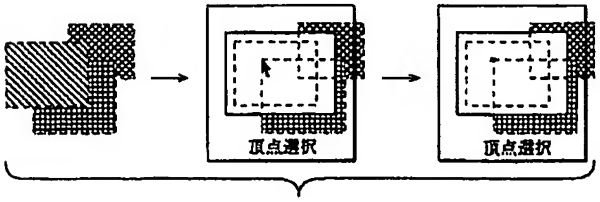
【図29】



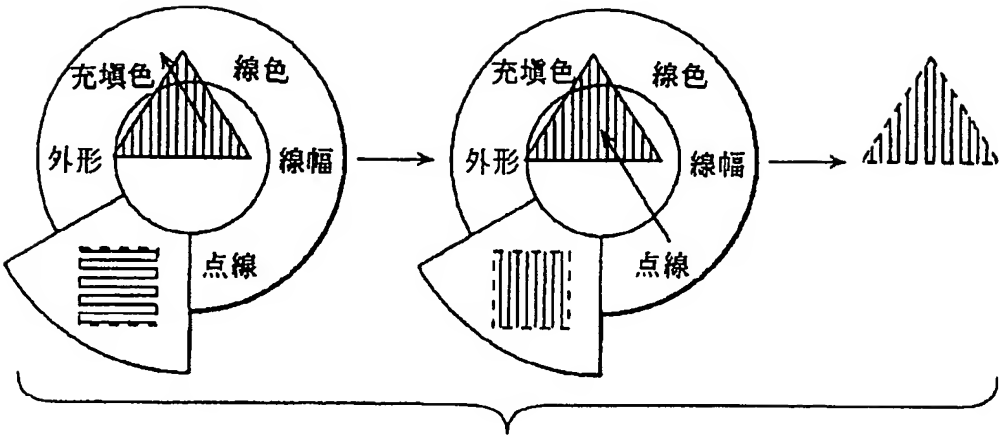

【図11】



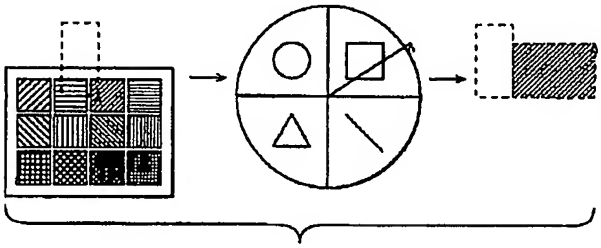
【図12】



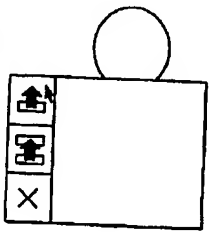
【図１３】



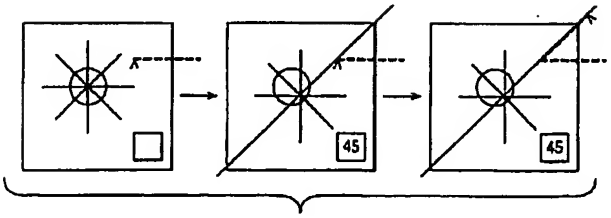

【図１４】



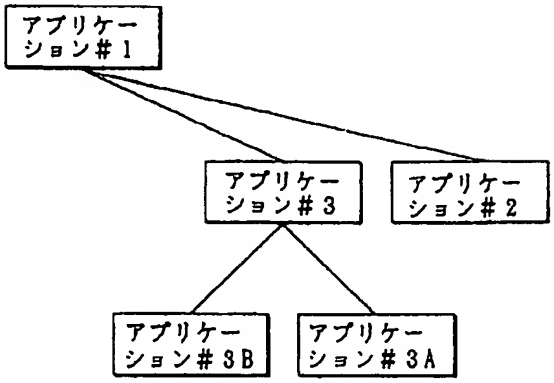
【図３１】



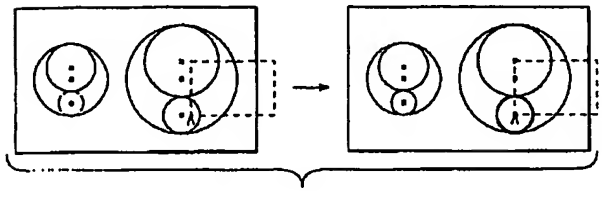
【図１５】



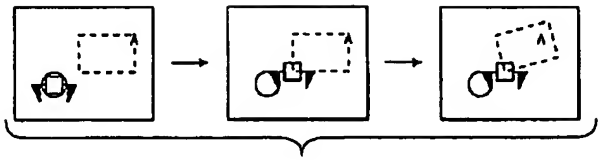

【図４０】



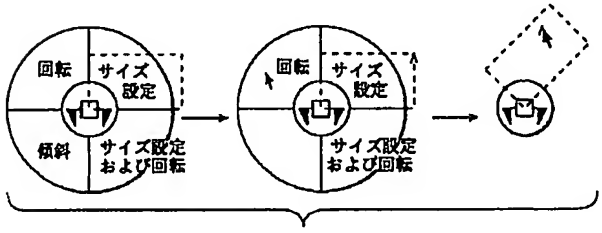

【図１６】



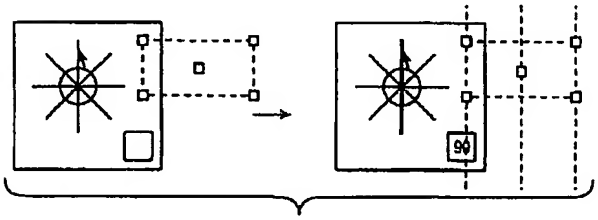
【図１７】



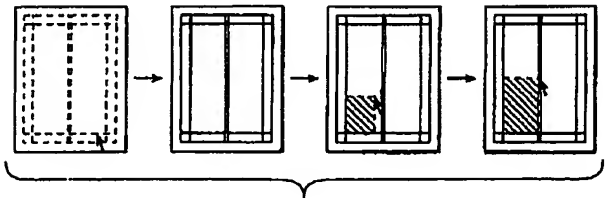
【図１８】



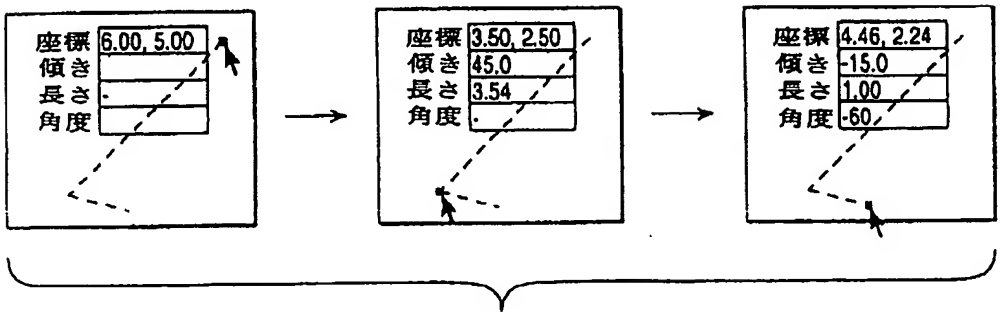

【図１９】



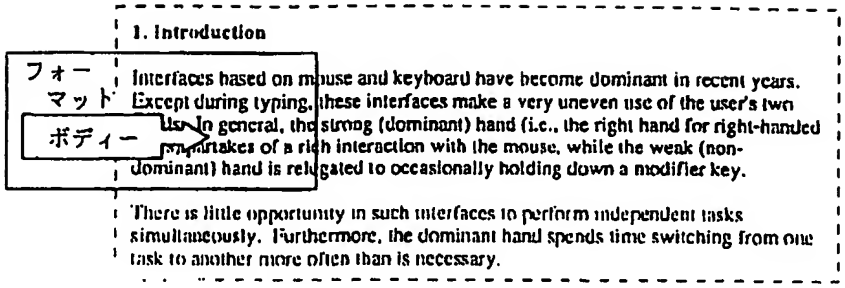

【図２１】



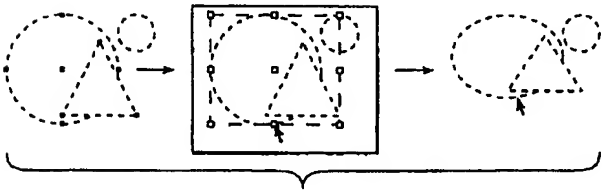
【図２２】



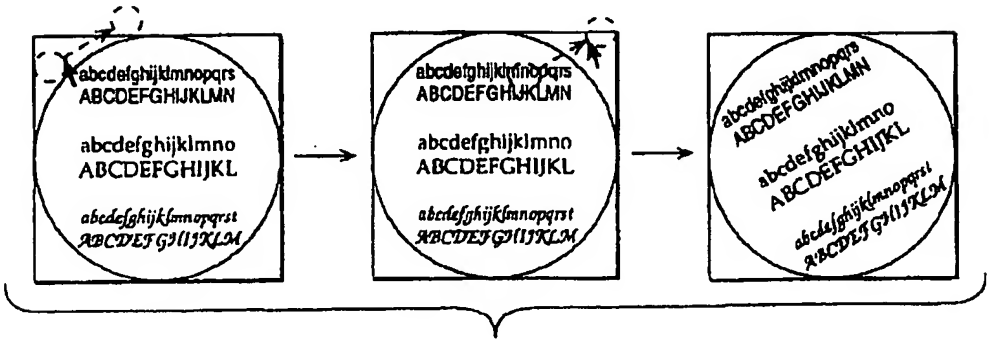
【図２３】



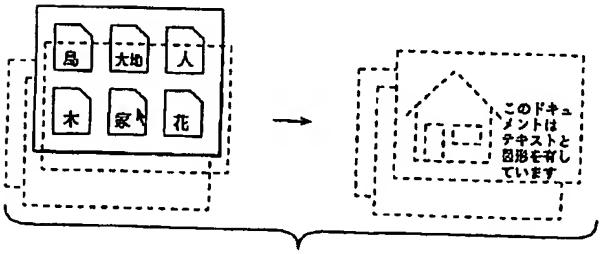

【図２５】



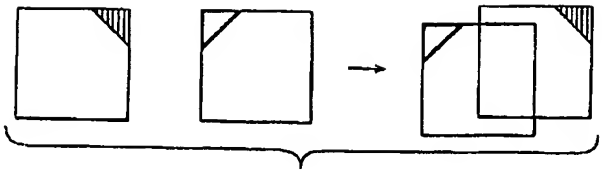
【図２８】



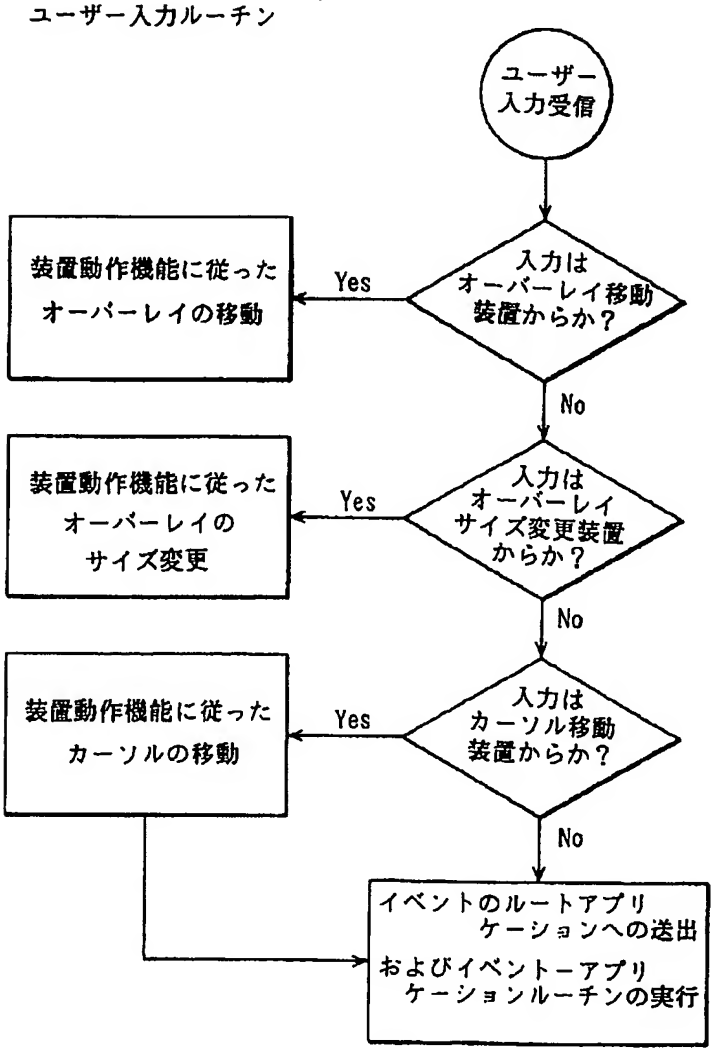

【図３０】



【図３２】

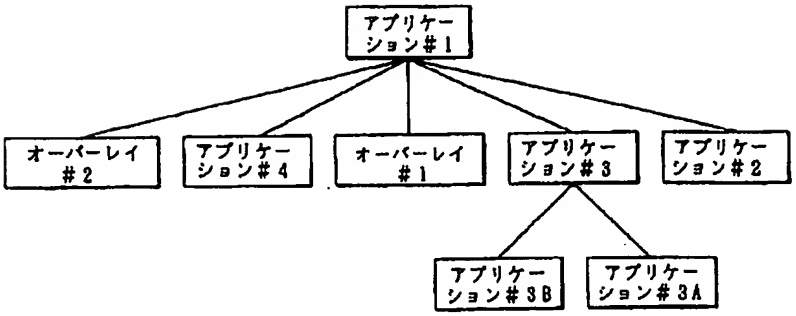


【図３３】

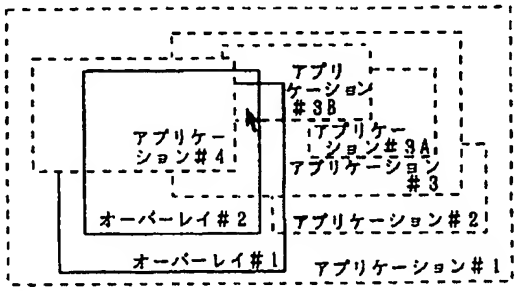

ユーザー入力ルーチン
ユーザー入力受信
入力はオーバーレイ移動装置からか？
Yes
装置動作機能に従ったオーバーレイの移動
No
入力はオーバーレイサイズ変更装置からか？
Yes
装置動作機能に従ったオーバーレイのサイズ変更
No
入力はカーソル移動装置からか？
Yes
装置動作機能に従ったカーソルの移動
No
イベントのルートアプリケーションへの送出
およびイベントーアプリケーションルーチンの実行

【図３４】

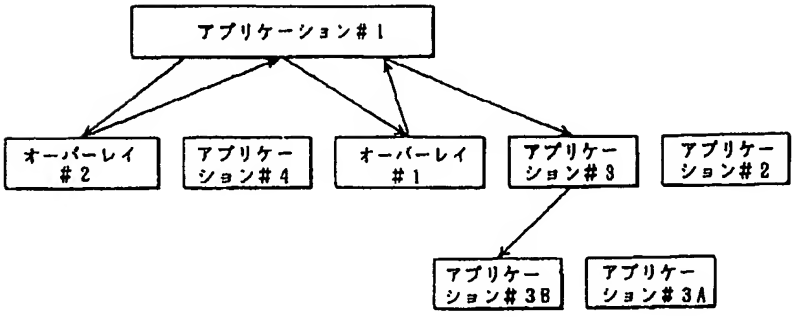
(A)


アプリケーション＃１
オーバーレイ＃２
アプリケーション＃４
オーバーレイ＃１
アプリケーション＃３
アプリケーション＃２
アプリケーション＃３Ｂ
アプリケーション＃３Ａ

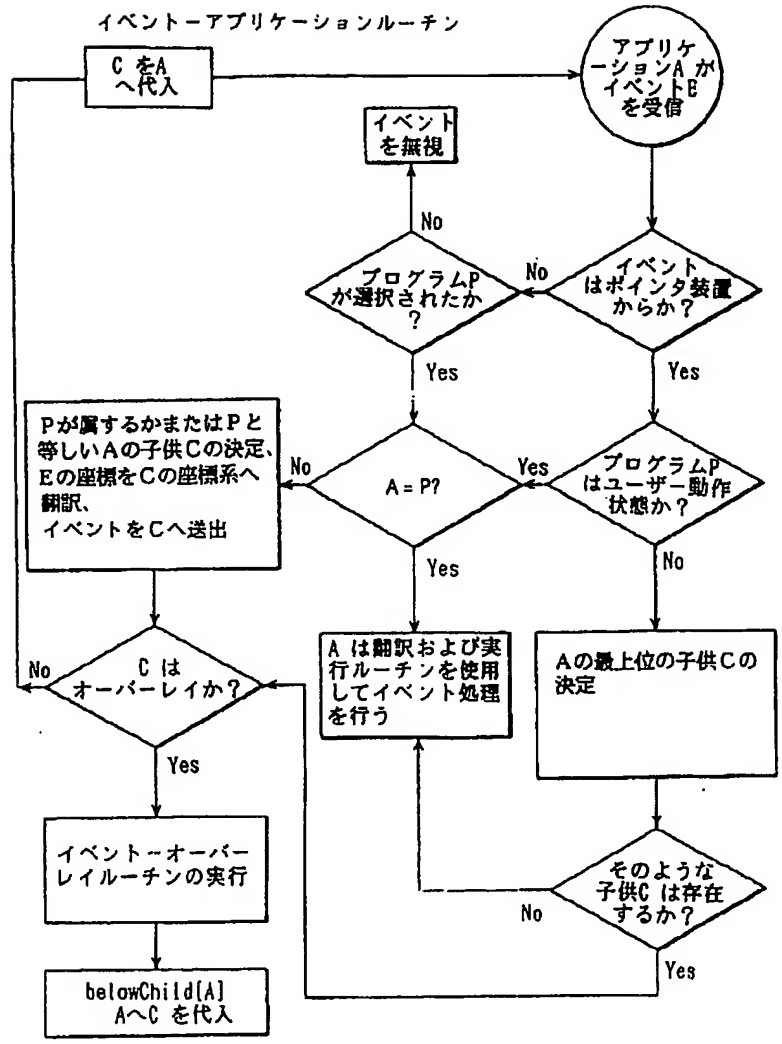
(B)


アプリケーション＃３Ｂ
アプリケーション＃４
アプリケーション＃３Ａ
アプリケーション＃３
オーバーレイ＃２
アプリケーション＃２
オーバーレイ＃１
アプリケーション＃１

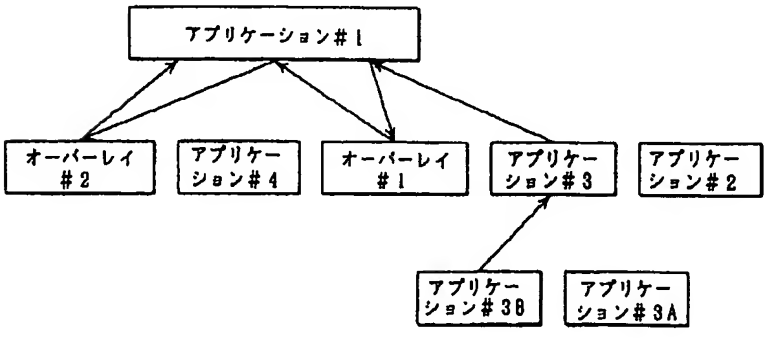
(C)


アプリケーション＃１
オーバーレイ＃２
アプリケーション＃４
オーバーレイ＃１
アプリケーション＃３
アプリケーション＃２
アプリケーション＃３Ｂ
アプリケーション＃３Ａ

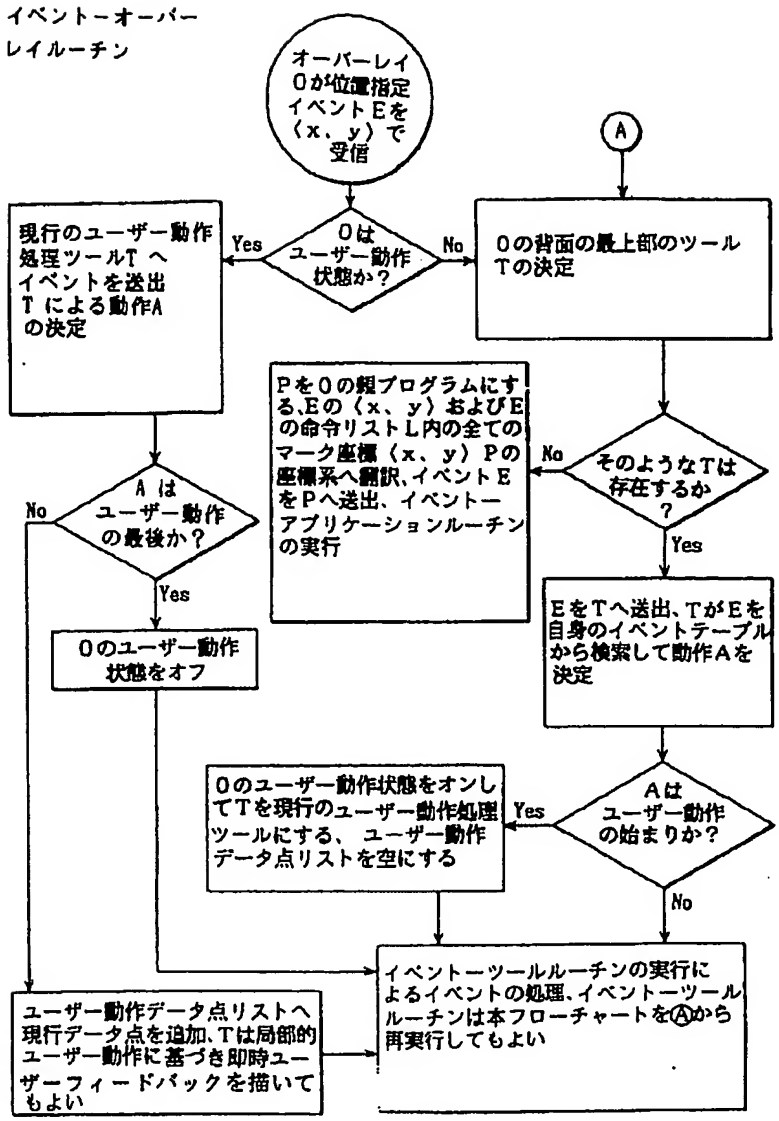
【図35】


イベントーアプリケーションルーチン
C をA へ代入
アプリケーションA がイベントE を受信
イベントを無視
No
プログラムPが選択されたか？
No
イベントはポインタ装置からか？
Yes
Yes
Pが属するかまたはPと等しいAの子供Cの決定、Eの座標をCの座標系へ翻訳、イベントをCへ送出
No
A = P?
Yes
プログラムPはユーザー動作状態か？
Yes
No
No
C はオーバーレイか？
A は翻訳および実行ルーチンを使用してイベント処理を行う
Aの最上位の子供Cの決定
Yes
イベントーオーバーレイルーチンの実行
そのような子供C は存在するか？
No
Yes
belowChild[A] AへC を代入

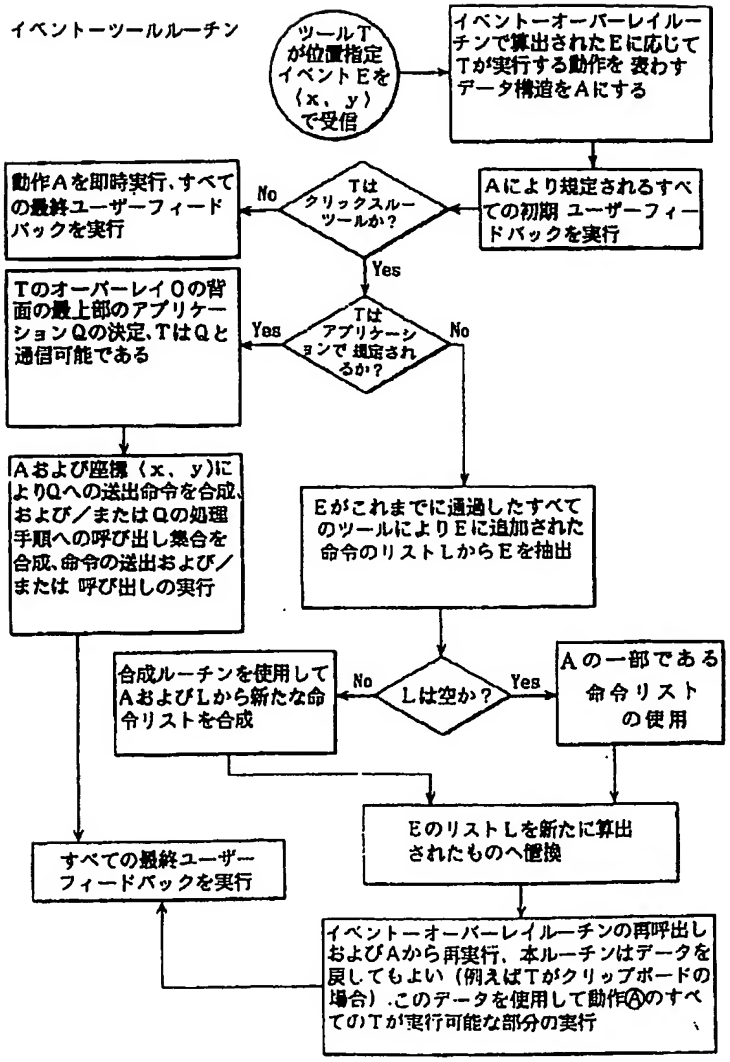
【図39】


アプリケーション#1
オーバーレイ#2
アプリケーション#4
オーバーレイ#1
アプリケーション#3
アプリケーション#2
アプリケーション#3B
アプリケーション#3A

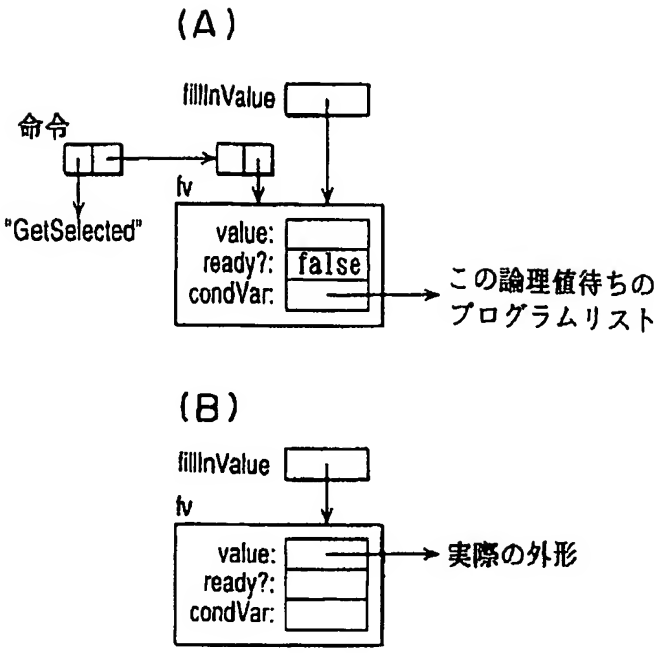
【図36】


イベントーオーバーレイルーチン
オーバーレイOが位置指定イベントEを〈x、y〉で受信
Oはユーザー動作状態か？
Yes
No
現行のユーザー動作処理ツールTへイベントを送出Tによる動作Aの決定
Oの背面の最上部のツールTの決定
A
Aはユーザー動作の最後か？
No
Yes
Oのユーザー動作状態をオフ
PをOの親プログラムにする、Eの〈x、y〉およびEの命令リストL内の全てのマーク座標〈x、y〉Pの座標系へ翻訳、イベントEをPへ送出、イベントーアプリケーションルーチンの実行
そのようなTは存在するか？
No
Yes
EをTへ送出、TがEを自身のイベントテーブルから検索して動作Aを決定
Aはユーザー動作の始まりか？
Yes
No
Oのユーザー動作状態をオンしてTを現行のユーザー動作処理ツールにする、ユーザー動作データ点リストを空にする
ユーザー動作データ点リストへ現行データ点を追加、Tは局部的ユーザー動作に基づき即時ユーザーフィードバックを描いてもよい
イベントーツールルーチンの実行によるイベントの処理、イベントーツールルーチンは本フローチャートをAから再実行してもよい

【図３７】


イベントーツールルーチン
ツールTが位置指定イベントEを〈x、y〉で受信
イベントーオーバーレイルーチンで算出されたEに応じてTが実行する動作を表わすデータ構造をAにする
Aにより規定されるすべての初期 ユーザーフィードバックを実行
Tはクリックスルーツールか？
No
動作Aを即時実行、すべての最終ユーザーフィードバックを実行
Yes
Tはアプリケーションで規定されるか？
Yes
Tのオーバーレイ0の背面の最上部のアプリケーションQの決定、TはQと通信可能である
No
Aおよび座標〈x、y〉によりQへの送出命令を合成、および／またはQの処理手順への呼び出し集合を合成、命令の送出および／または 呼び出しの実行
Eがこれまでに通過したすべてのツールによりEに追加された命令のリストLからEを抽出
Lは空か？
No
合成ルーチンを使用してAおよびLから新たな命令リストを合成
Yes
Aの一部である命令リストの使用
Eのリストしを新たに算出されたものへ置換
すべての最終ユーザーフィードバックを実行
イベントーオーバーレイルーチンの再呼出しおよびAから再実行、本ルーチンはデータを戻してもよい（例えばTがクリップボードの場合）.このデータを使用して動作Ⓐのすべてのが実行可能な部分の実行

【図38】




フロントページの続き

(72)発明者　ウィリアム　バクストン
カナダ国　エム４エル　１エックス６　オンタリオ州トロント　ケルンス　アヴェニュー　83

(72)発明者　モリーン　シー．ストーン
アメリカ合衆国　94022　カリフォルニア州　ロス　アルトス　パイン　レーン　191